Package 'gwas.lasso'

December 10, 2018

Type Package

Title Bayesian Lasso and Group Lasso for GWAS

Version 0.5

Date 2018-12-10

Author Zhong Wang

Maintainer Zhong Wang <wzhy2000@hotmail.com>

Description Bayesian Group Lasso for GWAS

License GPL (>= 2)

Depends R (>= 2.10.0)

Imports nlme, snpStats, scales, snowfall

Suggests

NeedsCompilation yes

Repository CRAN

Date/Publication 2018-12-10 07:21:49

R topics documented:

bls.plink 2
bls.simple 5
bls.simulate 7
bls.snpmat 9
gls.plink 11
gls.simple 16
gls.simulate 18
gls.snpmat 20
plot.BLS.ret 23
plot.GLS.ret 24
print.sum.BLS.ret 25
print.sum.GLS.ret 26
summary.BLS.ret 26
summary.GLS.ret 27

Index **29**

`bls.plink` *Scan PLINK dataset by the Bayesian lasso model*

Description

Scan PLINK dataset with single measurement and detect the joint additive and dominant effects by the Bayesian lasso Model(BLS Model).

Usage

```
bls.plink(file.phe,
      file.plink.bed, file.plink.bim, file.plink.fam,
      Y.name,
      covar.names,
      refit = TRUE,
      add.used = TRUE,
      dom.used = TRUE,
      fgwas.filter = FALSE,
      options = NULL,
      force.split=FALSE,
      plink.command=NULL)
```

Arguments

`file.phe`	Input CSV file containing one single measurement and covariate data.
`file.plink.bed`	Input PLINK data file, a binary file containing genotype information.
`file.plink.bim`	Input PLINK data file, an extended map file.
`file.plink.fam`	Input PLINK data file, family information.
`Y.name`	String indicating column name of response variable.
`covar.names`	String or vector of string indicating column names of covariates.
`refit`	Logical, if TRUE, the refit procedure should be processed. Default: TRUE.
`add.used`	Logical, if TRUE, additive effects will be estimated jointly. Default: TRUE.
`dom.used`	Logical, if TRUE, dominant effects will be estimated jointly. Default: TRUE.
`fgwas.filter`	Logical, if TRUE, the SNPs will be filtered by fGWAS model before the LASSO method is applied. Default: FALSE.
`options`	A list containing control parameters, including `nParallel.cpu`, `nMcmcIter`, `nPiecewise.ratio`, `fBurnInRound`, `fRhoTuning`, `fgwas.cutoff`, and debug.See blow details.
`force.split`	Logical, if TRUE, the PLINK command will be applied to extract the data for each single chromosome and then the analysis will be performed on these single chromosomes separately. Default: FALSE.
`plink.command`	String, the full path of PLINK command, if not assigned, data extraction maybe failed due to PLINK command can not be found from defulat path.

Details

(1) PLINK binary dataset
The PLINK binary dataset includes three files, *bed*,*bim* and *fam* file.
The *bed* file is a compressed binary file containing genotype information.
The *bim* file is an extended map file where each line of this file defines a SNP information containing exactly 4 columns: chromosome, SNP identifier, genetic distance and base-pair position (bp units).
The *fam* file is a family file containing the individuals and phenotypic information including six columns: Family ID, Individual ID, Paternal ID, Maternal ID, Sex (1=male; 2=female; other=unknown) and phenotype.
This package does **not** use the phenotype information stored in the fam file. The Phenotype values and covariates are stored into a CSV file which should be filled with numerical values and missing values encoded as NA.

(2) fGWAS model
A SNP filter strategy named fGWAS is proposed in this package. This method is used to detect the marginal genetic effects. If the parameter `fgwas.filter` is TRUE, the fGWAS method will be employed to remove the non-significant SNPs. Once SNPs are filtered by the fGWAS method, significant SNPs are remained to do lasso process.

(3) Variable selection and refit
A two-stage procedure based on Bayesian lasso or Group lasso is proposed, including variable selection and refit process. It is necessary to refit the lasso model after variable selection, where only selected SNPs are included in the final process. The parameter `refit` indicates whether to carry out the refit function.

(4) Control parameters
The control parameters include the following items.

1	nParallel.cpu	Default:1, the number of the CPU used to calculate.
2	nMcmcIter	Default:2000, the number of iteration of the MCMC algorithm.
3	BurnInRound	Default:0.3, a ratio to cut off unstable iterations in order to obtain the reasonable results.
4	RhoTuning	Default:0.095, a ratio used for narrowing the scope of parameter rho to reduce parameter oscillation.
5	fQval.add	Default:0.025, quantile of additive effect for each SNP, indicating how the confidence interval is calculated, e.g. range is 2.5-97.5% if fQval.add is 0.025.
6	fQval.dom	Default:0.025, quantile of dominant effect for each SNP.
7	fgwas.cutoff	Default:0.05, a p-value cutoff for fGWAS model to select the significant marginal SNPs.
8	nPiecewise.ratio	Default:2, the ratio of grouping SNPs.
9	debug	Default:FALSE, logical value, indicating whether or not debug information is outputted.

Value

An object of class `BLS.ret` including the following elements.

`varsel` Matrix, additive and dominant effects estimated by the procedure of variable selection for each SNP. Eleven columns are available,

- (1) chromosome group
- (2) position
- (3) significant sign of additive effect, 1: siginificant, 0:nonsignificant.
- (4) mean value of additive effect
- (5) minimum value of addtive effect
- (6) maximum value of addtive effect
- (7) significant sign of dominant effect, 1: siginificant, 0:nonsignificant.
- (8) mean value of dominant effect
- (9) minimum value of dominant effect
- (10) maximum value of dominant effect
- (11) Heritability rate

`varsel_cov` Matrix, covariate effects estimated by the variable selection procedure, Four columns as follows:

- (1) significant sign of covariate effect, 1: siginificant, 0:nonsignificant.
- (2) overall mean value of covariate effect
- (3) minimum value of covariate effect
- (4) maximum value of covariate effect

`refit` Matrix, additive and dominant effects estimated by the refit procedure for the SNPs selected by the variable selection, Eleven columns are same as the matrix of varible selection(`varsel`).

`refit_cov` Matrix, covariate effects estimated by the refit procedure, 4 columns are same as the matrix of varible selection(`varsel_cov`).

`fgwas.filter` Matrix, the p-values estimated by fGWAS method for each SNP, 5 columns as follows:

- (1) SNP ID
- (2) chromosome group
- (3) position
- (4) likelihood ratio
- (5) p-value

`options` List, data file names and all calling parameters, including the control parameters defined in `options`

Examples

```
bls.phe.out <- "bls.test.plink.phe"
bls.snp.out <- "bls.test.plink.snp"

sigsnp <- c(1:5)*5;

r.sim <- bls.simulate( bls.phe.out, bls.snp.out,
                simu_grp=1, simu_n= 600, simu_p=50,
                simu_snp_rho = 0.1,
```

```
                simu_rho     = 0.4,
                simu_sigma2  = 9,
                simu_mu      = 24,
                simu_cov_range=c( 0, 1),
                simu_cov_effect = c( 0, 2 ),
                simu_add_pos   = c( sigsnp[1], sigsnp[2], sigsnp[3]),
                simu_add_effect= c( 2.2, -2.5, 2.0 ),
                simu_dom_pos   = c( sigsnp[3], sigsnp[4], sigsnp[5]),
                simu_dom_effect= c( 2.8, 2.0, -2.5 ),
                simu_t_range = c(-1, 1),
                plink.format = TRUE,
                debug = FALSE );

r.bls <- bls.plink( bls.phe.out,
                r.sim$file.plink.bed,
                r.sim$file.plink.bim,
                r.sim$file.plink.fam,
                Y.name="Y",
                covar.names=c("X_1", "X_2"),
                fgwas.filter=FALSE );

summary(r.bls);

plot(r.bls);
```

`bls.simple` *Scan simple format dataset by the Bayesian lasso model*

Description

Scan simple format dataset with single measurement and detect the joint additive and dominant effects by the Bayesian lasso model (BLS model).

Usage

```
bls.simple(file.phe,
      file.snp,
      Y.name,
      covar.names,
      refit=TRUE,
      add.used=TRUE,
      dom.used=TRUE,
      fgwas.filter=FALSE,
      options=NULL)
```

Arguments

`file.phe`	Input CSV file containing one single measurement and covariate data.
`file.snp`	Input CSV file containing chromosome, position and genotypes of SNPs.
`Y.name`	String indicating the column name of response variable.

covar.names	String or vector of string indicating the column names of covariates.
refit	Logical, if TRUE, the refit procedure should be processed. Default: TRUE.
add.used	Logical, if TRUE, the additive effects will be estimated jointly. Default: TRUE.
dom.used	Logical, if TRUE, the dominant effects will be estimated jointly. Default: TRUE.
fgwas.filter	Logical, If TRUE, the SNPs will be filtered by fGWAS model before the lasso method is applied. Default: FALSE.
options	A list containing the control parameters, including nParallel.cpu, nMcmcIter, nPiecewise.ratio, fBurnInRound, fRhoTuning, fgwas.cutoff, and debug. See "Details" in the bls.plink.

Details

Simple format dataset
The simple format is a user-defined format which is designed to store small amount of SNPs for the users who do not use PLINK.
The genotypic data are stored in the CSV format, where each line describes a single SNP and must start with 2 columns of chromosome information (chromosome number and SNP position).Three genotypes (aa=0,Aa=1,AA=2) and missing data (coded as -1 or NA) are valid SNP values.

Value

An object of class BLS.ret, see "Details" in the bls.plink.

References

(1)Beyasian lasso model
Li, J., Das, K., Fu, G., Li, R., & Wu, R. (2011). The Bayesian lasso for genome-wide association studies. Bioinformatics, 27(4), 516-523.
(2)fGWAS model
Li, J., Wang, Z., Li, Y., & Wu, R. (2010). Functional genome-wide association studies of longitudinal traits. Handbook of Adaptive Designs in Pharmaceutical and Clinical Development.

Examples

```
# Set the filenames for simple format dataset.
bls.phe.out <- "bls.test.simple.phe";
bls.snp.out <- "bls.test.simple.snp";
# Makea simulation data using the defalut parameters.
bls.simulate(bls.phe.out, bls.snp.out,simu_n= 500, simu_p=50,
            simu_add_pos=c( 10, 20, 30), simu_dom_pos=c( 30, 40, 50));
# Call Beyasian lasso model to estimate the joint effects.
r.bls <- bls.simple(bls.phe.out,bls.snp.out,
      Y.name="Y",
      covar.names=c(),
      refit=FALSE,
```

```
      options=list(nPiecewise.ratio=0));

# List the significant information detcted by the BLS model
summary(r.bls);
# Plot the data
plot(r.bls);
```

`bls.simulate` *Generate simulation data of the Beyasian lasso model*

Description

Generate the simulation dataset of the Beyasian lasso model(BLS model).

Usage

```
bls.simulate(file.phe.out,
      file.snp.out,
      simu_grp = 1,
      simu_n = 500,
      simu_p =1000,
      simu_snp_rho = 0.1,
      simu_snp_missing = 0.002,
      simu_rho = 0.4,
      simu_sigma2 = 3,
      simu_mu = 26,
      simu_cov_range = c(0,1),
      simu_cov_effect= c( 0, 2 ),
      simu_add_pos = c( 100, 200, 300),
      simu_add_effect= c(2.2,-2.5,2.0),
      simu_dom_pos = c(300, 500, 700),
      simu_dom_effect= c(2.8,2.0,-2.5),
      simu_t_range = c(-1,1),
      plink.format = FALSE,
      debug = FALSE)
```

Arguments

`file.phe.out`	String, the name of output phenotypic data file.
`file.snp.out`	String, the name of output genotypic data file.
`simu_grp`	Integer, the number of groups that SNPs divided. Default: 1.
`simu_n`	Integer, sample size. Default: 500.
`simu_p`	Integer, number of SNPs. Default: 1000.
`simu_snp_rho`	Float, the correlation coefficient between two adjacent SNPs. Default: 0.1.

`simu_snp_missing`
: Float, the ratio of missing SNPs.
Default: 0.002.

`simu_rho`
: Float, the correlation coefficient between two adjacent time points.
Default: 0.4.

`simu_sigma2`
: Float, the variance of AR(1) convariance matrix.
Default: 3.

`simu_mu`
: Float, overall mean of phenotypic data.
Default: 26.

`simu_cov_range`
: Vector, range of covariate values.
Default: c(0,1).

`simu_cov_effect`
: Vector indicating coefficient of the covariates.
Default: c(0,2), means two covariates which the effects are 0 and 2.

`simu_add_pos`
: Vector, indicating positions of the significant SNPs with additive effects.
Default: c(100, 200, 300).

`simu_add_effect`
: Vector, indicating additive effects of significant SNPs.
Default: c(2.2, -2.5, 2).

`simu_dom_pos`
: Vector, indicating positions of the significant SNPs with dominant effects.
Default: c(300, 500, 700).

`simu_dom_effect`
: Vector, dominant effects of significant SNPs.
Default: c(2.8, 2, -2.5).

`simu_t_range`
: Vector, range of time points.
Default: c(-1, 1)

`plink.format`
: Logical, if TRUE, save the SNP data as PLINK format, otherwise, save as simple format.
Default: FALSE.

`debug`
: Logical, if TRUE, run the command in debug model which inputs more messages for debugging.
Default: FALSE.

Details

The function uses the pre-defined parameters to create a simulation dataset. Two data formats are supported in this function, one is PLINK format, which functions are provided by snpStats package, another is simple-format, which is created by this package.

For PLINK format, 4 files are generated. The *bed* file, *bim* file and *fam* file can be accessed by PLINK command or the snpStats package.

For simple format, 2 files are generated. The genotypic data file is encoded 2, 1, 0, NA to indicate genotypes QQ, Qq, qq and missing data respectively. The phenotypic file and genotypic file can be viewed by any text editor.

It is assumed that the trait is controlled by some SNPs, and the positions of these SNPs are specified by the parameter `simu_a_pos` and `simu_d_pos`. The genetic effects of overall mean and causal

SNPs are specified by the parameter `simu_mu`, `simu_a_effect` and `simu_d_effect`.

Except genetic effects, covariates can be simulated in this function. The covariate effects are specified by parameter `simu_covar_effect`.

Value

If PLINK format is specified, 4 filenames, including phenotype file,*bed* file, *bim* file and *fam* file, are returned in a list object. Otherwise, two simple format filenames, the phenotypic data file and the genotypic data file, are returned.

Examples

```
# Create the Simple format dataset by defulat parameter
bls.simulate("bls.simple.phe", "bls.simple.snp");

# Create the PLINK dataset by defulat parameter
bls.simulate("bls.simple.phe", "bls.simple.snp", plink.format=TRUE);
```

---

bls.snpmat *Scan matrix dataset by the Bayesian lasso model*

Description

Scan matrix dataset with single measurement and detect the joint additive and dominant effects by the Bayesian lasso model (BLS model).

Usage

```
bls.snpmat(phe.mat,
      snp.mat,
      Y.name,
      covar.names,
      refit = TRUE,
      add.used = TRUE,
      dom.used = TRUE,
      fgwas.filter = FALSE,
      options = NULL)
```

Arguments

`phe.mat`	Matrix, phenotypic data containing one single measurement and covariate data.
`snp.mat`	Matrix, genotypic data containing chromosome, position and genotypes of SNPs.
`Y.name`	String, the column name of response variable.
`covar.names`	String or vector of string indicating the column names of covariates.
`refit`	Logical, if TRUE, the refit procedure should be processed. Default: TRUE.

add.used	Logical, if TRUE, the additive effects will be estimated jointly. Default: TRUE.
dom.used	Logical, if TRUE, the dominant effects will be estimated jointly. Default: TRUE.
fgwas.filter	Logical, If TRUE, the SNPs will be filtered by fGWAS model before the lasso method is applied. Default: FALSE.
options	A list containing control parameters, including nParallel.cpu, nMcmcIter, nPiecewise.ratio, fBurnInRound, fRhoTuning, fgwas.cutoff, and debug. See details in bls.plink.

Details

The phenotypic data and genotypic data are stored in the matrix. The first two columns in the genotypic matrix must be chromosome and position information and the other columns are SNP data encode by 0,1,2 and NA.

The phenotypic matrix inludes the following columns:
1) Individual ID
2) Response varible(Y)
3) One or more covariates(X, X_1,...

Value

An object of class BLS.ret, see "Details" in the bls.plink.

References

(1)Beyasian lasso model
Li, J., Das, K., Fu, G., Li, R., & Wu, R. (2011). The Bayesian lasso for genome-wide association studies. Bioinformatics, 27(4), 516-523.
(2)fGWAS model
Li, J., Wang, Z., Li, Y., & Wu, R. (2010). Functional genome-wide association studies of longitudinal traits. Handbook of Adaptive Designs in Pharmaceutical and Clinical Development.

Examples

```
# Generate the simulation data set.
bls.phe.out <- "bls.test.simple.phe";
bls.snp.out <- "bls.test.simple.snp";
r.sim <- bls.simulate( bls.phe.out, bls.snp.out,simu_n= 500, simu_p=100,
            simu_add_pos=c( 10, 20, 30), simu_dom_pos=c( 30, 40, 50));

# Load the phenotypic data into a matrix
tb.phe<-read.csv(bls.phe.out, header=TRUE);
# Set IDs as the row name
rownames(tb.phe) <- tb.phe[,1];
tb.phe <- tb.phe[,-1];

# Load the genotypic data into a matrix
tb.snp<-read.csv(bls.snp.out);
```

```
# Call Beyasian lasso model to estimate the joint effects.
r.bls <- bls.snpmat(tb.phe, tb.snp,
      Y.name="Y",
      covar.names=c("X_1","X_2"),
      fgwas.filter = FALSE );

# Show the signiciant SNPs and effects.
summary(r.bls);

# Plot the data
plot(r.bls);
```

`gls.plink` *Scan PLINK dataset by the Group lasso model*

Description

Scan PLINK dataset with longitudinal measurements and detect the joint additive and dominant effects by the Group lasso model (GLS model).

Usage

```
gls.plink(file.phe,
      file.plink.bed,file.plink.bim,file.plink.fam,
      Y.prefix,
      Z.prefix,
      covar.names,
      refit = TRUE,
      add.used = TRUE,
      dom.used = TRUE,
      fgwas.filter = FALSE,
      gpu.used = FALSE,
      options = NULL,
      force.split=FALSE,
      plink.command=NULL)
```

Arguments

`file.phe`	Input CSV file containing longitudinal measurements and covariate data.
`file.plink.bed`	Input PLINK data file, a binary file containing genotype information.
`file.plink.bim`	Input PLINK data file, an extended map file.
`file.plink.fam`	Input PLINK data file, family information.
`Y.prefix`	String, the prefix of column names of response variables.
`Z.prefix`	String, the prefix of column names of measurement times.
`covar.names`	String or vector of string indicating column names of covariates.
`refit`	Logical, if TRUE, the refit procedure should be processed. Default: TRUE.

add.used	Logical, if TRUE, the additive effects will be estimated jointly. Default: TRUE.
dom.used	Logical, if TRUE, the dominant effects will be estimated jointly. Default: TRUE.
fgwas.filter	Logical, If TRUE, the SNPs will be filtered by fGWAS model before the lasso method is applied. Default: FALSE.
gpu.used	Logical, if TRUE, the computational performance will be improved by GPU if GPU is avalaible. Default: FALSE.
options	A list containing control parameters, including nLegendre, nParallel.cpu, nMcmcIter, nPiecewise.ratio, fBurnInRound, fRhoTuning, fgwas.cutoff, fQval.add, fQval.dom, and debug. See the details in bls.plink.
force.split	Logical, if TRUE, the PLINK command will be applied to extract the data for each single chromosome and then the analysis will be performed on these single chromosomes separately. Default: FALSE.
plink.command	String, the full path of PLINK command, if not assigned, data extraction maybe failed due to PLINK command can not be found from defulat path.

Details

(1) PLINK dataset
The PLINK dataset includes three files, *bed*,*bim* and *fam* file.
The *bed* file is a compressed binary file containing genotype information.
The *bim* file is an extended map file where each line of this file defines a SNP information containing exactly 4 columns: chromosome, SNP identifier, genetic distance and base-pair position (bp units).
The *fam* file is a family file containing the individuals and phenotypic information including six columns: Family ID, Individual ID, Paternal ID, Maternal ID, Sex (1=male; 2=female; other=unknown) and phenotype.
This package does **not** use the phenotype information stored in the fam file. The phenotype values and covariates are stored into a CSV file which should be filled with numerical values and missing values encoded as NA.

(2) fGWAS model
A SNP filter strategy named fGWAS is proposed in this package. This method is used to detect the marginal genetic effects. If the parameter fgwas.filter is true, the fGWAS method will be employed to remove the nonsignificant SNPs. Once SNPs are filtered by the fGWAS method, significant SNPs are remained to do LASSO process.

(3) Variable selection and refit
A two-stage procedure based on Bayesian Lasso or Group Lasso is proposed, including variable selection and refit process. It is necessary to refit the lasso model after variable selection, where only selected SNPs are included in the refit process. The parameter refit indicates whether to carry out the refit function.

(4) Control parameters
The control parameters include the following items.

- nParallel.cpu Default:0, the number of the CPU used to calculate.
- nPiecewise.ratio Default:2, the ratio of grouping SNPs.
- nMcmcIter Default:2000, the number of iteration of the MCMC algorithm execution.
- fBurnInRound Default:0.3, a ratio to cut off unstable iteration in order to make the result precise.
- fRhoTuning Default:0.095, a ratio used for narrowing the scope of parameter rho to reduce parameter oscillation.
- fgwas.cutoff Default:0.05, a p-value cutoff for fGWAS model to select the significant marginal SNPs.
- debug Default:FALSE, logical value, indicating whether or not debug information is outputted.

Value

An object of class GLS.ret including the following elements.

varsel_add Matrix, **additive** effects estimated by the procedure of variable selection for the SNPs selected by fGWAS filter. Twenty-one columns are available,

- (1) chromosome group
- (2) position
- (3) the sign of **1st** Legendre polynomial of **additive** effcts
- (4) the sign of **2nd** Legendre polynomial of **additive** effcts
- (5) the sign of **3rd** Legendre polynomial of **additive** effcts
- (6) the sign of **4th** Legendre polynomial of **additive** effcts
- (7) the L2 norm of mean values of **additive** effects, i,e, L=sqrt(add_1^2 + add_2^2 + add_3^2 + add_4^2)
- (8) the mean value of **1st** Legendre coefficient of **additive** effects in the range of (fQval.add, 1 - fQval.add) of sorted steable iteration.
- (9) the mean value of **2nd** Legendre coefficient of **additive** effects in the range of (fQval.add, 1 - fQval.add) of sorted steable iteration.
- (10) the mean value of **3rd** Legendre coefficient of **additive** effects in the range of (fQval.add, 1 - fQval.add) of sorted steable iteration.
- (11) the mean value of **4th** Legendre coefficient of **additive** effects in the range of (fQval.add, 1 - fQval.add) of sorted steable iteration.
- (12) the L2 norm of minimum values of **additive** effects
- (13) the minimum value of **1st** Legendre coefficient of **additive** effects in the range of (fQval.add, 1 - fQval.add) of sorted steable iteration.
- (14) the minimum value of **2nd** Legendre coefficient of **additive** effects in the range of (fQval.add, 1 - fQval.add) of sorted steable iteration.
- (15) the minimum value of **3rd** Legendre coefficient of **additive** effects in the range of (fQval.add, 1 - fQval.add) of sorted steable iteration.
- (16) the minimum value of **4th** Legendre coefficient of **additive** effects in the range of (fQval.add, 1 - fQval.add) of sorted steable iteration.
- (17) the L2 norm of maximum values of **additive** effects
- (18) the maximum value of **1st** Legendre coefficient of **additive** effects in the range of (fQval.add, 1 - fQval.add) of sorted steable iteration.
- (19) the maximum value of **2nd** Legendre coefficient of **additive** effects in the range of (fQval.add, 1 - fQval.add) of sorted steable iteration.

- (20) the maximum value of **3rd** Legendre coefficient of **additive** effects in the range of (fQval.add, 1 - fQval.add) of sorted steable iteration.
- (21) the maximum value of **4th** Legendre coefficient of **additive** effects in the range of (fQval.add, 1 - fQval.add) of sorted steable iteration.

`varsel_dom` Matrix, **dominant** effects estimated by the procedure of variable selection for the SNPs selected by fGWAS filter. 21 columns are same as `varsel_add`.

`varsel_cov` Matrix, covariate effects estimated by the variable selection procedure, 19 columns as follows:

- (1) the sign of **1st** Legendre polynomial of **covariate** effcts
- (2) the sign of **2nd** Legendre polynomial of **covariate** effcts
- (3) the sign of **3rd** Legendre polynomial of **covariate** effcts
- (4) the sign of **4th** Legendre polynomial of **covariate** effcts
- (5) the L2 norm of mean values of **covariate** effects, i,e, L2=sqrt(add_1^2 + add_2^2 + add_3^2 + add_4^2)
- (6) the mean value of **1st** Legendre coefficient of **covariate** effects in the range of (fQval.add, 1 - fQval.add) of sorted steable iteration.
- (7) the mean value of **2nd** Legendre coefficient of **covariate** effects in the range of (fQval.add, 1 - fQval.add) of sorted steable iteration.
- (8) the mean value of **3rd** Legendre coefficient of **covariate** effects in the range of (fQval.add, 1 - fQval.add) of sorted steable iteration.
- (9) the mean value of **4th** Legendre coefficient of **covariate** effects in the range of (fQval.add, 1 - fQval.add) of sorted steable iteration.
- (10) the L2 norm of minimum values of **covariate** effects
- (11) the minimum value of **1st** Legendre coefficient of **covariate** effects in the range of (fQval.add, 1 - fQval.add) of sorted steable iteration.
- (12) the minimum value of **2nd** Legendre coefficient of **covariate** effects in the range of (fQval.add, 1 - fQval.add) of sorted steable iteration.
- (13) the minimum value of **3rd** Legendre coefficient of **covariate** effects in the range of (fQval.add, 1 - fQval.add) of sorted steable iteration.
- (14) the minimum value of **4th** Legendre coefficient of **covariate** effects in the range of (fQval.add, 1 - fQval.add) of sorted steable iteration.
- (15) the L2 norm of maximum values of **covariate** effects
- (16) the maximum value of **1st** Legendre coefficient of **covariate** effects in the range of (fQval.add, 1 - fQval.add) of sorted steable iteration.
- (17) the maximum value of **2nd** Legendre coefficient of **covariate** effects in the range of (fQval.add, 1 - fQval.add) of sorted steable iteration.
- (18) the maximum value of **3rd** Legendre coefficient of **covariate** effects in the range of (fQval.add, 1 - fQval.add) of sorted steable iteration.
- (19) the maximum value of **4th** Legendre coefficient of **covariate** effects in the range of (fQval.add, 1 - fQval.add) of sorted steable iteration.

`refit_add` Matrix, **additive** effects estimated by the refit procedure for the SNPs selected by variable selection, 21 columns are same as the matrix of varible selection. See also (`varsel_add`).

`refit_dom` Matrix, **dominant** effects estimated by the refit procedure for the SNPs selected by variable selection, 21 columns are same as the matrix of varible selection. See also (`varsel_dom`).

`refit_cov` Matrix, **covariate** effects estimated by the refit procedure for the SNPs selected by variable selection, 19 columns are same as the matrix of varible selection. See also (`varsel_cov`).

`fgwas` Matrix, the p-values estimated by fGWAS method for each SNP, 5 columns as follows:

- (1) SNP ID
- (2) chromosome group
- (3) position
- (4) likelihood ratio
- (5) p-value

`varsel_PSRF` Matrix, the potential scale reduction factors in the variable selection procedure.

`refit_PSRF` Matrix, the potential scale reduction factors in the refit procedure.

`options` List, data file names and all parameters, including the control parameters defined in `options`

References

(1)Group lasso model
Li, J., Wang, Z., Li, R., & Wu, R. (2015). Bayesian group Lasso for nonparametric varying-coefficient models with application to functional genome-wide association studies. The Annals of Applied Statistics, 9(2), 640-664.
(2)fGWAS model
Li, J., Wang, Z., Li, Y., & Wu, R. (2010). Functional genome-wide association studies of longitudinal traits. Handbook of Adaptive Designs in Pharmaceutical and Clinical Development.

Examples

```
# Set the filename of simulation data set
gls.phe.out <- "gls.test.simple.phe"
gls.snp.out <- "gls.test.simple.snp"

# Set the additive effects for 3 SNPs
a_effect <- array(0, dim=c(3,4));
a_effect[1,]<-c( 1.04, 0.885, -2.055, 0.545);
a_effect[2,]<-c( 1.17, -0.20, 0.74, -4.715);
a_effect[3,]<-c( 1.40, -2.25, 1.00,  0.00);

# Set the dominant effects for 3 SNPs
d_effect <- array(0, dim=c(3,4));
d_effect[1,]<-c( 1.49, -2.135, 4.82, 1.425);
d_effect[2,]<-c( 1.045, 1.320, 1.905,  1.535);
d_effect[3,]<-c( 1.265, -1.225, 2.710, -1.96);

# Set the coefficient for 2 covariates
cov_effect <- array(0, dim=c(2,4));
cov_effect[1,]<-c( 2.49, -1.135, 0.82, 0.425);
cov_effect[2,]<-c( -1.045, 2.320, 0.905,  0.535);

#Set the significant positions
sigsnp <- c(1, 3, 5,7,9);

#Create simulation dataset.
r.sim <- gls.simulate( gls.phe.out, gls.snp.out,
                simu_n= 400, simu_grp=1, simu_p=10,
                simu_snp_rho=0.4, simu_rho=0.1, simu_sigma2= 4,
```

```
                simu_mu= c(13.395, -3.08, 1.875, -3.195),
                simu_cov_effect = cov_effect,
                simu_cov_range  = c(-1,1),
                simu_add_pos    = c( sigsnp[1], sigsnp[2], sigsnp[3] ),
                simu_add_effect = a_effect,
                simu_dom_pos    = c( sigsnp[3], sigsnp[4], sigsnp[5] ),
                simu_dom_effect = d_effect,
                simu_z_range    = c(30,60), simu_z_count = c(5,12),
                plink.format=TRUE,
                debug=FALSE);

# Call theGroup model to estimate the genetic effects
r.gls <- gls.plink(gls.phe.out,
                r.sim$file.plink.bed,
                r.sim$file.plink.bim,
                r.sim$file.plink.fam,
                Y.prefix="Y",
                Z.prefix="Z",
                covar.names=c("X_1","X_2"),
                fgwas.filter = FALSE );


# Show the significant SNPs and effects
summary(r.gls);

# Plot the results
plot(r.gls);
```

`gls.simple` *Scan Simple format dataset by the Group lasso model*

Description

Scan Simple format dataset with longitudinal measurements and detect the joint additive and dominant effects by the Group lasso model (GLS model).

Usage

```
gls.simple(file.phe,
      file.snp,
      Y.prefix,
      Z.prefix,
      covar.names,
      refit = TRUE,
      add.used = TRUE,
      dom.used = TRUE,
      fgwas.filter = FALSE,
      gpu.used = FALSE,
      options = NULL)
```

Arguments

file.phe	Input CSV file containing longitudinal measurements and covariate data.
file.snp	Input CSV file containing chromosome, position and genotypes of SNPs.
Y.prefix	String, the prefix of column names of response variables.
Z.prefix	String, the prefix of column names of measure time.
covar.names	String or vector of string indicating the column names of covariates.
refit	Logical, if TRUE, the refit procedure should be processed. Default: TRUE.
add.used	Logical, if TRUE, additive effects will be estimated jointly. Default: TRUE.
dom.used	Logical, if TRUE, dominant effects will be estimated jointly. Default: TRUE.
fgwas.filter	Logical, If TRUE, the SNPs will be filtered by the fGWAS model before the lasso method is applied. Default: FALSE.
gpu.used	Logical, if TRUE, the computational performance will be improved by GPU if GPU is avalaible. Default: FALSE.
options	A list containing control parameters, including nLegendre, nParallel.cpu, nMcmcIter, nPiecewise.ratio, fBurnInRound, fRhoTuning, fgwas.cutoff, and debug. See the details in bls.plink.

Details

The simple format is introduced in the section of bls.simple. Other details can be found in the section of gls.plink

Value

An object of class GLS.ret, see "Details" in the gls.plink.

References

(1)Group lasso model
Li, J., Wang, Z., Li, R., & Wu, R. (2015). Bayesian group Lasso for nonparametric varying-coefficient models with application to functional genome-wide association studies. The Annals of Applied Statistics, 9(2), 640-664.
(2)fGWAS model
Li, J., Wang, Z., Li, Y., & Wu, R. (2010). Functional genome-wide association studies of longitudinal traits. Handbook of Adaptive Designs in Pharmaceutical and Clinical Development.

Examples

```
# Create simulation dataset using the default parameters
gls.phe.out <- "gls.test.simple.phe"
gls.snp.out <- "gls.test.simple.snp"

r.sim <- gls.simulate(gls.phe.out, gls.snp.out, simu_n=600, simu_p=10);
```

```
# Call the Grpup lasso model to estimate the effects.
r.gls <- gls.simple(gls.phe.out, gls.snp.out,
                Y.prefix="Y",
                Z.prefix="Z",
                covar.names="X",
                fgwas.filter = FALSE,
                options=list(nPiecewise.ratio=0) );

# Show the significant SNPs and effects,
summary(r.gls);

# Plot the results
plot(r.gls);
```

`gls.simulate` *Simulation data of the Group lasso model*

Description

Generating simulation data object by the Group lasso model (GLS model).

Usage

```
gls.simulate(file.phe.out,
      file.snp.out,
      simu_grp=1,
      simu_n=500,
      simu_p=1000,
      simu_snp_rho = 0.1,
      simu_snp_missing= 0.002,
      simu_rho = 0.4,
      simu_sigma2 = 16,
      simu_mu = c(13.395, -3.08, 1.875, -3.195),
      simu_cov_range = c( -1, 1 ),
      simu_cov_effect= array(c(0,0,0,0), dim=c(1,4)),
      simu_add_pos   = c( 1,2,3 ),
      simu_add_effect= array(c( 1.04, 0.885, -2.055, 0.545, 1.17,
      -0.20, 0.74, -4.715,1.40, -2.25, 1.00,  0.00), dim=c(3,4)),
      simu_dom_pos = c( 3,4,5 ),
      simu_dom_effect= array(c( 1.49, -2.135, 4.82, 1.425, 1.045,
      1.320, 1.905, 1.535, 1.265, -1.225, 2.710, -1.96),dim=c(3,4)),
      simu_z_range = c(20,80), simu_z_count = c( 5, 12 ),
      plink.format = FALSE,
      debug = FALSE )
```

Arguments

`file.phe.out`	String, the name of the output phenotypic data file.
`file.snp.out`	String, the name of the output genotypic data file.

`simu_grp` Numeric, the number of groups that snps are divided into for parallel computation.
Default: 1.

`simu_n` Integer, sample size.
Default: 500.

`simu_p` Integer, number of SNP.
Default: 1000.

`simu_snp_rho` Float, correlation coefficient between two adjacent SNPs.
Default: 0.1.

`simu_snp_missing`
Float, ratio of missing SNPs.

`simu_rho` Float, correlation coefficient between two adjacent time points.
Default: 0.4.

`simu_sigma2` Float, residual error.
Default: 16.

`simu_mu` Vector, Legendre coefficients of overall mean effect.
Default: c(13.395,-3.08, 1.875, -3.195).

`simu_cov_range` Vector, range of covariates.
Default: c(-1,1).

`simu_cov_effect`
Vector, Legendre coefficients of for covariates.
Default: array(c(0,0,0,0), dim=c(1,4).for 1 covariate.

`simu_add_pos` Vector, positions of the significant SNPs with additive effects.
Default: c(1,2,3).

`simu_add_effect`
Matrix, Legendre coefficients for additive effects of significant SNPs.
Default: array(c(1.04, 0.885, -2.055, 0.545, 1.17, -0.20, 0.74, -4.715, 1.40, -2.25, 1.00, 0.00), dim=c(3,4)).

`simu_dom_pos` Vector, positions of the significant SNPs with dominant effects. Default: c(3,4,5).

`simu_dom_effect`
Matrix, Legendre coefficients for dominant effects of significant SNPs.
Default: array(c(1.49, -2.135, 4.82, 1.425, 1.045, 1.320, 1.905, 1.535,1.265, -1.225, 2.710, -1.96), dim=c(3,4)).

`simu_z_range` Vector, range of measure times.
Default: c(20, 80).

`simu_z_count` Integer, the range of measure count.
Default: c(5, 12).

`plink.format` Logical, if TRUE, save the SNP data as PLINK format.
Default: FALSE.

`debug` Logical, if TRUE, run the command in debug model which inputs more messages for debugging.
Default: FALSE.

Details

The simulation in this package uses the pre-defined parameters to create a data object containing longitudinal phenotypic data and genotypic data.

The genotypic data file is coded 2, 1, 0, -1 to indicate genotypes QQ, Qq, qq and missing data respectively.

It is assumed that the trait is controlled by some SNPs jointly, and the positions of these SNPs will be generated by the parameters `simu_a_pos` and `simu_d_pos`. The genetic effects of overall mean and causal SNPs will be generated by the parameters `simu_mu`, `simu_a_effect` and `simu_d_effect`.

Given phenotypic data and genotype information, genetic effects of each SNPs could be estimated. However in GWAS a number of covariates either discrete or continuous may be measured for each subject. In the simulation function, covariate effects are generated by parameter `simu_covar_effect`.

In particular, since measurements within each subject are possibly correlated with one another, AR(1) model is employed to approximate the residual covariance matrix, and assume AR(1) with `simu_rho` and `simu_sigma2`.

Value

Returns two simulation data files: the phenotypic data file and genotypic data file.

References

Li, J., Wang, Z., Li, R., & Wu, R. (2015). Bayesian group Lasso for nonparametric varying-coefficient models with application to functional genome-wide association studies. The Annals of Applied Statistics, 9(2), 640-664.

Examples

```
gls.simulate( "gls.test.simple.phe", "gls.test.simple.snp", plink.format=TRUE );
```

---

gls.snpmat *scan matrix dataset by the Group lasso model*

Description

Scan matrix dataset with longitudinal measurements and detect the joint additive and dominant effects by the Group lasso model (GLS model).

Usage

```
gls.snpmat(phe.mat,
      snp.mat,
      Y.prefix,
      Z.prefix,
      covar.names,
      refit = TRUE,
```

```
        add.used = TRUE,
        dom.used = TRUE,
        fgwas.filter = FALSE,
        gpu.used=FALSE,
        options = NULL)
```

Arguments

phe.mat	Matrix, phenotypic data containing longitudinal measurements and covariate data.
snp.mat	Matrix, genotypic data containing chromosome, position and genotypes of SNPs.
Y.prefix	String, prefix of column names of response variables.
Z.prefix	String, prefix of column names of measurement times.
covar.names	String or vector of string indicating column names of covariates.
refit	Logical, if TRUE, the refit procedure should be processed. Default: TRUE.
add.used	Logical, if TRUE, additive effects will be estimated jointly. Default: TRUE.
dom.used	Logical, if TRUE, dominant effects will be estimated jointly. Default: TRUE.
fgwas.filter	Logical, If TRUE, the SNPs will be filtered by the fGWAS model before the lasso method is applied. Default: FALSE.
gpu.used	Logical, if TRUE, the computational performance will be improved by GPU if GPU is avalaible. Default: FALSE.
options	A list containing control parameters, including nLegendre, nParallel.cpu, nMcmcIter, nPiecewise.ratio, fBurnInRound, fRhoTuning, fgwas.cutoff, and debug. see details.

Details

(1) Matrix dataset
The phenotypic data and genotypic data are stored in the matrix format.

(2) fGWAS model
A SNP filter strategy named fGWAS is proposed in this package. This method is used to detect the marginal genetic effects. If the parameter fgwas.filter is true, the fGWAS method will be employed to remove the nonsignificant SNPs. Once SNPs are filtered by the fGWAS method, significant SNPs are remained to do LASSO process.

(3) Varible selection and refit
A two-stage procedure based on Bayesian Lasso or Group Lasso is proposed, including variable selection and refit process. It is necessary to refit the lasso model after variable selection, where only selected SNPs are included in the final process. The parameter refit indicates whether to carry out the refit function.

(4) Control parameters
The control parameters include the following items.

- nParallel.cpu Default:0, the number of the CPU used to calculate.
- nPiecewise.ratio Default:2, the ratio of grouping SNPs.
- nMcmcIter Default:2000, the number of iteration of the MCMC algorithm execution.
- fBurnInRound Default:0.3, a ratio to cut off unstable iteration in order to make the result precise.
- fRhoTuning Default:0.095, a ratio used for narrowing the scope of parameter rho to reduce parameter oscillation.
- fgwas.cutoff Default:0.05, a p-value cutoff for fGWAS model to select the significant marginal SNPs.
- debug Default:FALSE, logical value, indicating whether or not debug information is outputted.

Value

An object of class GLS.ret, see "Details" in the gls.plink.

References

(1)Group lasso model
Li, J., Wang, Z., Li, R., & Wu, R. (2015). Bayesian group Lasso for nonparametric varying-coefficient models with application to functional genome-wide association studies. The Annals of Applied Statistics, 9(2), 640-664.
(2)fGWAS model
Li, J., Wang, Z., Li, Y., & Wu, R. (2010). Functional genome-wide association studies of longitudinal traits. Handbook of Adaptive Designs in Pharmaceutical and Clinical Development.

Examples

```
gls.phe.out <- "gls.test.simple.phe";
gls.snp.out <- "gls.test.simple.snp";
r.sim <- gls.simulate( gls.phe.out, gls.snp.out, simu_n=400, simu_p=10)

tb.phe <- read.csv(gls.phe.out, header=TRUE);
# Set IDs as the row name
rownames(tb.phe) <- tb.phe[,1];
tb.phe <- tb.phe[,-1];

tb.snp <- read.csv(gls.snp.out);

r.gls <- gls.snpmat(tb.phe, tb.snp,
                Y.prefix="Y",
                Z.prefix="Z",
                covar.names="X",
                fgwas.filter = FALSE );

summary(r.gls);

plot(r.gls);
```

`plot.BLS.ret` *Plot the results of the BLS model*

Description

Plot a result object obtained from the BLS method.

Usage

```
## S3 method for class 'BLS.ret'
plot(x, y=NULL, ..., fig.prefix=NULL)
```

Arguments

`x`	An object obtained from `bls.simple` or `bls.plink`or `bls.snpmat`.
`y`	An null parameter, not used.
`...`	Other parameters.
`fig.prefix`	String, the prefix of output file name.

Details

The plot function can output three types of PDF figure, including:

1) The Manhattan figure exported by the fGWAS method (*.fgwas.pdf).

2) The genetic effects of all SNPs estimated by the variable selection procedure (*.varsel.pdf).

3) The genetic effects of significant SNPs estimated by the refit procedure (*.refit.pdf).

The Manhattan figure gives -log10(p-value) for each SNP. The variable selection only selects the SNPs with -log10(p-value) greater than the threshold value specified in the control parameters.

In the BLS model, the figures of genetic effects output heritability information.

References

Li, J., Das, K., Fu, G., Li, R., & Wu, R. (2011). The Bayesian lasso for genome-wide association studies. Bioinformatics, 27(4), 516-523.

See Also

See `bls.simple` or `bls.plink`or `bls.snpmat` to generate an `BLS.ret` object.

Examples

```
##r.bls is a result object obtained from bls.simpe, bls.plink, bls.snpmat
##e.g. r.bls <- bls.simple(...);
#
# plot(r.bls, fig.prefix="r.bls.pdf");
```

`plot.GLS.ret` *Plot the results of the GLS model*

Description

Plot a result object obtained by the GLS method.

Usage

```
## S3 method for class 'GLS.ret'
plot(x, y=NULL, ..., fig.prefix=NULL, q.probs = 0.1)
```

Arguments

`x`	An object obtained from `gls.simple` or `gls.plink`or `gls.snpmat`
`y`	An null parameter, not used.
`fig.prefix`	String, the prefix of output file name
`...`	Other parameters

Details

The plot function can output three types of PDF figure, including:

1) The Manhattan figure exported by the fGWAS method (*.fgwas.pdf).

2) The genetic effects of all SNPs estimated by the variable selection procedure (*.varsel.pdf).

3) The genetic effects of significant SNPs estimated by the refit procedure (*.refit.pdf).

The Manhattan figure gives -log10(p-value) for each SNP. The variable selection only selects the SNPs with -log10(p-value) greater than the threshold value specified in the control parameters.

In the GLS model, the figures of genetic effects will output the time-varying additive and dominant curves for each significant SNP.

References

Li, J., Wang, Z., Li, R., & Wu, R. (2015). Bayesian group Lasso for nonparametric varying-coefficient models with application to functional genome-wide association studies. The Annals of Applied Statistics, 9(2), 640-664.

See Also

See gls.simple or gls.plinkor gls.snpmat to generate an object

Examples

```
##r.gls is a result object obtained from gls.simpe, gls.plink, gls.snpmat
##e.g. r.gls <- gls.simple(...);
#
# plot(r.gls, fig.prefix="r.gls.pdf");
```

---

print.sum.BLS.ret *Print the results obtained from the BLS model*

Description

Print a result object obtained from the BLS method.

Usage

```
## S3 method for class 'sum.BLS.ret'
print(x, ...)
```

Arguments

x	An object obtained from bls.simple or bls.plinkor bls.snpmat
...	Other parameters

Details

The print command will print out the result object in the standard format.

References

Li, J., Das, K., Fu, G., Li, R., & Wu, R. (2011). The Bayesian lasso for genome-wide association studies. Bioinformatics, 27(4), 516-523.

See Also

See bls.simple or bls.plinkor bls.snpmat to generate an object

print.sum.GLS.ret *Print the result obtained from the GLS model*

Description

Print a result object obtained from the GLS method.

Usage

```
## S3 method for class 'sum.GLS.ret'
print(x, ...)
```

Arguments

x	An object obtained from gls.simple or gls.plink or gls.snpmat
...	Other parameters

Details

The print command will print out the result object in the standard format.

References

Li, J., Wang, Z., Li, R., & Wu, R. (2015). Bayesian group Lasso for nonparametric varying-coefficient models with application to functional genome-wide association studies. The Annals of Applied Statistics, 9(2), 640-664.

See Also

See gls.simple or gls.plinkor gls.snpmat to generate an object

summary.BLS.ret *Summarize the result obtained from the BLS model*

Description

Summarize the result object obtained from the BLS method.

Usage

```
## S3 method for class 'BLS.ret'
summary(object, ...)
```

Arguments

object	an object obtained by bls.simple or bls.plinkor bls.snpmat
...	Other parameters

Details

The summary command summarizes some tables of the result object r.bls, and the command exports all of this values to R console.

References

Li, J., Das, K., Fu, G., Li, R., & Wu, R. (2011). The Bayesian lasso for genome-wide association studies. Bioinformatics, 27(4), 516-523.

See Also

See bls.simple or bls.plinkor bls.snpmat to generate an object

Examples

```
##r.bls is a result object obtained from bls.simpe, bls.plink, bls.snpmat
##e.g. r.bls <- bls.simple(...);
#
# summary(r.bls);
```

---

summary.GLS.ret *Summarize the result obtained from the GLS model*

Description

Summarize the result object obtained from the GLS method.

Usage

```
## S3 method for class 'GLS.ret'
summary(object, ...)
```

Arguments

object	an object obtained from gls.simple or gls.plinkor gls.snpmat
...	Other parameters

Details

The summary command summarizes some tables of the result object r.gls, and the command exports all of this values to R console.

References

Li, J., Wang, Z., Li, R., & Wu, R. (2015). Bayesian group Lasso for nonparametric varying-coefficient models with application to functional genome-wide association studies. The Annals of Applied Statistics, 9(2), 640-664.

Examples

```
##r.gls is a result object obtained from gls.simpe, gls.plink, gls.snpmat
##e.g. r.gls <- gls.simple(...);
#
# summary(r.gls);
```

Index

∗Topic **BLS model**
bls.plink, 2
bls.simple, 5
bls.simulate, 7
bls.snpmat, 9
plot.BLS.ret, 23
print.sum.BLS.ret, 25
summary.BLS.ret, 26
∗Topic **GLS model**
gls.plink, 11
gls.simple, 16
gls.simulate, 18
gls.snpmat, 20
plot.GLS.ret, 24
print.sum.GLS.ret, 26
summary.GLS.ret, 27
∗Topic **PLINK**
bls.plink, 2
gls.plink, 11
∗Topic **functions**
bls.plink, 2
bls.simple, 5
bls.simulate, 7
bls.snpmat, 9
gls.plink, 11
gls.simple, 16
gls.simulate, 18
gls.snpmat, 20
∗Topic **plot**
plot.BLS.ret, 23
plot.GLS.ret, 24
∗Topic **print**
print.sum.BLS.ret, 25
print.sum.GLS.ret, 26
∗Topic **simple**
bls.simple, 5
gls.simple, 16
∗Topic **simulate**
bls.simulate, 7
gls.simulate, 18
∗Topic **summary**
summary.BLS.ret, 26
summary.GLS.ret, 27

bls.plink, 2, *6*, *10*, *12*, *17*, *23*, *25–27*
bls.simple, 5, *17*, *23*, *25–27*
bls.simulate, 7
bls.snpmat, 9, *23*, *25–27*

gls.plink, 11, *17*, *22*, *24–27*
gls.simple, 16, *24–27*
gls.simulate, 18
gls.snpmat, 20, *24–27*

plot.BLS.ret, 23
plot.GLS.ret, 24
print.sum.BLS.ret, 25
print.sum.GLS.ret, 26

summary.BLS.ret, 26
summary.GLS.ret, 27