Package 'KMgene'

November 22, 2017

Type Package

Title Gene-Based Association Analysis for Complex Traits

Version 1.2

Author Qi Yan

Maintainer Qi Yan <qiy17@pitt.edu>

Description Gene based association test between a group of SNPs and traits including familial (both continuous and binary) traits, multivariate continuous (both independent and familial) traits, longitudinal continuous traits and survival traits. Part of methods were previously published in Maity et al (2012) <doi:10.1002/gepi.21663>, Chen et al (2013) <doi:10.1002/gepi.21703>, Chen et al (2014) <doi:10.1002/gepi.21791>, Yan et al (2015) <doi:10.1159/000375409>, Yan et al (2015) <doi:10.1534/genetics.115.178590> and Yan et al (2015) <doi:10.1159/000445057>.

License GPL (>= 2)

Encoding UTF-8

LazyData true

Depends CompQuadForm, MASS, nlme, kinship2

Imports mgcv, coxme, survival, Matrix

RoxygenNote 6.0.0

Collate 'saddle.R' 'pchisqsum2.R' 'CoxKM.R' 'Data_CoxKM_data.R' 'Data_FKM_charID.R' 'Data_FKM_numID.R' 'Data_FbKM_charID.R' 'Data_FbKM_numID.R' 'Data_LKM_charID.R' 'Data_LKM_numID.R' 'Data_LKM_numID_int.R' 'Data_MFKM_charID.R' 'Data_MFKM_numID.R' 'Data_MKM_charID.R' 'Data_MKM_numID.R' 'FKM.R' 'SKAT_Optimal_Integrate_Func_Davies.R' 'Get_Liu_Params_Mod.R' 'FKMO.R' 'FKMO_Null_Model.R' 'FKM_Null_Model.R' 'FbKM.R' 'FbKMO.R' 'glmmPQL.R' 'FbKMO_Null_Model.R' 'FbKM_Null_Model.R' 'LKM.R' 'LKMO.R' 'LKMO_Null_Model.R' 'LKM_Int.R' 'LKM_Null_Model.R' 'MFKM.R' 'MFKMO.R' 'MFKMO_Null_Model.R' 'MFKM_Null_Model.R' 'MKM.R' 'MKMO.R' 'MKMO_Null_Model.R' 'MKM_Null_Model.R' 'c.KMgene.R' 'coxlr.fit.R' 'ginv_s.R' 'prepCoxKM.R'

NeedsCompilation yes

Repository CRAN

Date/Publication 2017-11-22 20:27:41 UTC

R topics documented:

CoxKM *KM for Survival Traits in GWAS Data (calculate p-value)*

Description

This function (CoxKM) is used to perform the KM analysis (Chen et al., 2014) for survival traits in GWAS data

Usage

```
CoxKM(..., SNPInfo = NULL, wts = function(maf) {     dbeta(maf, 1, 25) },
  method = "saddlepoint", snpNames = "Name", aggregateBy = "gene",
  mafRange = c(0, 0.5), verbose = FALSE)
```

Arguments

`...`	Object saved from prepCoxKM.
`SNPInfo`	The SNP Info file. This should contain the fields listed in snpNames and aggregateBy. Only SNPs in this table will be meta analyzed, so this may be used to restrict the analysis.
`wts`	Either a function to calculate testing weights, or a character specifying a vector of weights in the SNPInfo file. For skatMeta the default are the 'beta' weights.
`method`	The p-value calculation method. Default is "saddlepoint", "integration" is the Davies method used in the SKAT package.
`snpNames`	The field of SNPInfo where the SNP identifiers are found. Default is "Name"
`aggregateBy`	The field of SNPInfo on which the skat results were aggregated. Default is "gene".
`mafRange`	Range of MAF's to include in the analysis (endpoints included). Default is all SNPs (0 <= MAF <= 0.5).
`verbose`	Logical. Whether progress bars should be printed.

Value

output: survival trait KM (CoxKM) p-value

Examples

```
############################################################
### Examples for Survival Traits in GWAS Data using KM ###
############################################################
data("CoxKM_data")
library(survival)
cov <- as.matrix(pheno1[,2:3])
p<-vector()
for (i in 1:2) {
  Gene <- unique(SNPInfo$gene)[i]
```

```
  choose <- SNPInfo$gene == Gene
  geno <- Z1[,choose]
  cohort1 <- prepCoxKM(Z=Z1, Surv(time, status)~strata(sex)+bmi,
  SNPInfo=SNPInfo[choose,], data=pheno1)
  p[i] <- CoxKM(cohort1, SNPInfo=SNPInfo[choose,])
}
```

CoxKM_data *This is survival data*

Description

data.frame

Usage

```
data(CoxKM_data)
```

Details

- pheno1. sex, bmi, time and status for 600 samples
- Z1. genotypes for 600 samples for two genes
- SNPInfo. SNP name and gene name

FbKM *KM for Binary Traits in Familial GWAS Data (calculate p-value)*

Description

This function (FbKM) is used to perform the KM analysis (Yan et al., 2014) for binary traits in familial GWAS data

Usage

```
FbKM(obj, genotypes, gid, weights = NULL, acc = 1e-04,
  append.write = NULL)
```

Arguments

obj	results saved from FbKM_Null_Model.
genotypes	1st column: gene name; 2nd column: snp name; 3rd-end columns: A matrix of genotypes for each subject (class: data.frame). The order of 3rd-end columns should match id. Coded as 0, 1, 2 and no missing. This genotype file can be a big file containing all genes or it can be files containing one single gene.

gid	A vector of id mapping to samples in genotype file (class: vector). So the order of samples in gid must be the same as the order in genotypes. Make sure it is not a factor. Although gid doesn't have to be in the same order as id, it is suggested to make them sorted in the same order in order to make all files easily to be tracked. No missing.
weights	1st column: gene name; 2nd column: snp name; 3rd column: A vector with the length equal to the number of variants in the test (class: data.frame). Default is Null indicating equal weight for all markers
acc	Accuracy of numerical integration used in Davies' method. Default 1e-4.
append.write	The name of pvalue output file. Write out p-values in real time. Don't need to wait until all genes are processed to get the final output.

Value

output: binary trait family KM (Fb-KM) p-value

Examples

```
#################################################################
### Examples for Binary Traits in Familial GWAS Data using KM ###
#################################################################
### Subject IDs are numeric ###
data("FbKM_numID")
obj1 <- FbKM_Null_Model(phenotype=fbkm_n_y$y, id=fbkm_n_y$id, fa=fbkm_n_y$fa,
mo=fbkm_n_y$mo, family="binomial", covariates=NULL)
pvalue1 <- FbKM(obj=obj1, genotypes=fbkm_n_gene, gid=fbkm_n_geneid$gid, weights=NULL,
append.write="./pvalues.out")
pvalue1 <- FbKM(obj=obj1, genotypes=fbkm_n_gene, gid=fbkm_n_geneid$gid, weights=NULL,
append.write=NULL)
obj2 <- FbKM_Null_Model(phenotype=fbkm_n_y$y, id=fbkm_n_y$id, fa=fbkm_n_y$fa,
mo=fbkm_n_y$mo, family="binomial", covariates=NULL)
pvalue2 <- FbKM(obj=obj2, genotypes=fbkm_n_gene, gid=fbkm_n_geneid$gid, weights=
fbkm_n_weights, append.write=NULL)
obj3 <- FbKM_Null_Model(phenotype=fbkm_n_y$y, id=fbkm_n_y$id, fa=fbkm_n_y$fa,
mo=fbkm_n_y$mo, family="binomial", covariates=fbkm_n_covariates)
pvalue3 <- FbKM(obj=obj3, genotypes=fbkm_n_gene, gid=fbkm_n_geneid$gid, weights=
fbkm_n_weights, append.write=NULL)
# Read in a list of genes files instead of a big file containing all genes
obj4 <- FbKM_Null_Model(phenotype=fbkm_n_y$y, id=fbkm_n_y$id, fa=fbkm_n_y$fa,
mo=fbkm_n_y$mo, family="binomial", covariates=NULL)
gene <- split(fbkm_n_gene, fbkm_n_gene[,1])
for (k in 1:2) {
  gene[[k]]$gene <- as.character(gene[[k]]$gene)
  pvalue1 <- FbKM(obj=obj4, genotypes=gene[[k]], gid=fbkm_n_geneid$gid, weights=NULL,
  append.write="./pvalues.out")
  }
### Subject IDs are character ###
data("FbKM_charID")
obj1 <- FbKM_Null_Model(phenotype=fbkm_c_y$y, id=as.character(fbkm_c_y$id),
fa=as.character(fbkm_c_y$fa), mo=as.character(fbkm_c_y$mo), family="binomial",
covariates=NULL)
```

```
pvalue1 <- FbKM(obj=obj1, genotypes=fbkm_c_gene, gid=as.character(fbkm_c_geneid$gid),
weights=NULL)
ys <- fbkm_c_y[order(fbkm_c_y$id),] #Sorted character IDs
obj1 <- FbKM_Null_Model(phenotype=ys$y, id=as.character(ys$id), fa=as.character(ys$fa),
mo=as.character(ys$mo),  covariates=NULL)
pvalue1 <- FbKM(obj=obj1, genotypes=fbkm_c_gene, gid=as.character(fbkm_c_geneid$gid),
weights=NULL)
```

`FbKMO` *Optimal KM for Non-continuous Traits in Familial GWAS Data (calculate p-value)*

Description

This function (FbKMO) is used to perform the optimal KM analysis for binary traits in familial GWAS data

Usage

```
FbKMO(obj, genotypes, gid, weights = NULL, acc = 1e-04, acc2 = 1e-04,
  r.all = c(0, 0.25, 0.5, 0.75, 1), append.write = NULL)
```

Arguments

`obj`	results saved from FbKMO_Null_Model.
`genotypes`	1st column: gene name; 2nd column: snp name; 3rd-end columns: A matrix of genotypes for each subject (class: data.frame). The order of 3rd-end columns should match id. Coded as 0, 1, 2 and no missing. This genotype file can be a big file containing all genes or it can be files containing one single gene.
`gid`	A vector of id mapping to samples in genotype file (class: vector). So the order of samples in gid must be the same as the order in genotypes. Make sure it is not a factor. Although gid doesn't have to be in the same order as id, it is suggested to make them sorted in the same order in order to make all files easily to be tracked. No missing.
`weights`	1st column: gene name; 2nd column: snp name; 3rd column: A vector with the length equal to the number of variants in the test (class: data.frame). Default is Null indicating equal weight for all markers
`acc`	Accuracy of numerical integration used in Davies' method for individual r.all p-values. Default 1e-4.
`acc2`	Accuracy of numerical integration used in Davies' method for the final p-value. Default 1e-4.
`r.all`	A list of predefined proportion of linear kernel and burden test. When r.all=0, regular kernel machine test (FbKM); when r.all=1, burden test.
`append.write`	The name of pvalue output file. Write out p-values in real time. Don't need to wait until all genes are processed to get the final output.

Value

output: binary trait optimal family KM (Fb-KMO) p-value

Examples

```
#########################################################################
### Examples for Binary Traits in Familial GWAS Data using optimal KM ###
#########################################################################
### Subject IDs are numeric ###
data("FbKM_numID")
obj1 <- FbKMO_Null_Model(phenotype=fbkm_n_y$y, id=fbkm_n_y$id, fa=fbkm_n_y$fa,
mo=fbkm_n_y$mo, family="binomial", covariates=NULL)
pvalue1 <- FbKMO(obj=obj1, genotypes=fbkm_n_gene, gid=fbkm_n_geneid$gid, weights=NULL,
append.write="./pvalues.out")
# Read in a list of genes files instead of a big file containing all genes
obj <- FbKMO_Null_Model(phenotype=fbkm_n_y$y, id=fbkm_n_y$id, fa=fbkm_n_y$fa,
mo=fbkm_n_y$mo, family="binomial", covariates=NULL)
gene <- split(fbkm_n_gene, fbkm_n_gene[,1])
for (k in 1:2) {
  gene[[k]]$gene <- as.character(gene[[k]]$gene)
  pvalue1 <- FbKMO(obj=obj, genotypes=gene[[k]], gid=fbkm_n_geneid$gid, weights=NULL,
  append.write="./pvalues.out")
}
### Subject IDs are character ###
data("FbKM_charID")
obj1 <- FbKMO_Null_Model(phenotype=fbkm_c_y$y, id=as.character(fbkm_c_y$id),
fa=as.character(fbkm_c_y$fa), mo=as.character(fbkm_c_y$mo), family="binomial",
covariates=NULL)
pvalue1 <- FbKMO(obj=obj1, genotypes=fbkm_c_gene, gid=as.character(fbkm_c_geneid$gid),
weights=NULL)
```

FbKMO_Null_Model *Optimal KM for Binary Traits in Familial GWAS Data (fit null model)*

Description

This function (FbKMO) is used to perform optimal KM analysis for binary traits in familial GWAS data
In the final correlation matrix, the covariance between parent and offspring is 0.5, the covariance between siblings is also 0.5

Usage

```
FbKMO_Null_Model(phenotype, id, fa, mo, family = "binomial",
  covariates = NULL)
```

Arguments

phenotype	A vector of quantitative trait in the analysis (class: vector). The order should match the vector id. Subjects with missing phenotypes are only used for kinship calculation.
id	A vector of id (class: vector). It can be either numeric or character. The id indicates each subject. Make sure it is not factor. No missing.
fa	A vector of father id (class: vector). It can be either numeric or character. The father id indicates the father of each subject. If this subject has no father in this data, the value is set to "NA". Make sure it is not factor.
mo	A vector of mother id (class: vector). It can be either numeric or character. The mother id indicates the mother of each subject. If this subject has no mother in this data, the value is set to "NA". Make sure it is not factor.
family	Type of phenotype. (Default="binomial")
covariates	A matrix of covariates (class: data.frame). The order of rows should match the vector id. Default NULL. Subjects with missing covariates are only used for kinship calculation.

Value

output: object as input for FbKMO

FbKM_charID *This is binary family data with character IDs*

Description

data.frame

Usage

```
data(FbKM_charID)
```

Details

- fbkm_c_covariates. age and sex for 100 samples
- fbkm_c_gene. genotypes for 100 samples for two genes and each gene has 50 SNPs
- fbkm_c_geneid. ordered sample id as in fbkm_c_gene (100 samples)
- fbkm_c_weights. weight for each SNP
- fbkm_c_y. sample id, sample's father id, sample's mother id and trait

FbKM_Null_Model *KM for Binary Traits in Familial GWAS Data (fit null model)*

Description

This function (FbKM) is used to perform KM analysis (Yan et al., 2015) for binary traits in familial GWAS data
In the final correlation matrix, the covariance between parent and offspring is 0.5, the covariance between siblings is also 0.5

Usage

```
FbKM_Null_Model(phenotype, id, fa, mo, family = "binomial",
  covariates = NULL)
```

Arguments

phenotype	A vector of quantitative trait in the analysis (class: vector). The order should match the vector id. Subjects with missing phenotypes are only used for kinship calculation.
id	A vector of id (class: vector). It can be either numeric or character. The id indicates each subject. Make sure it is not factor. No missing.
fa	A vector of father id (class: vector). It can be either numeric or character. The father id indicates the father of each subject. If this subject has no father in this data, the value is set to "NA". Make sure it is not factor.
mo	A vector of mother id (class: vector). It can be either numeric or character. The mother id indicates the mother of each subject. If this subject has no mother in this data, the value is set to "NA". Make sure it is not factor.
family	Type of phenotype. (Default="binomial")
covariates	A matrix of covariates (class: data.frame). The order of rows should match the vector id. Default NULL. Subjects with missing covariates are only used for kinship calculation.

Value

output: object as input for FbKM

FbKM_numID *This is binary family data numeric IDs*

Description

data.frame

Usage

```
data(FbKM_numID)
```

Details

- fbkm_n_covariates. age and sex for 100 samples
- fbkm_n_gene. genotypes for 100 samples for two genes and each gene has 50 SNPs
- fbkm_n_geneid. ordered sample id as in fbkm_n_gene (100 samples)
- fbkm_n_weights. weight for each SNP
- fbkm_n_y. sample id, sample's father id, sample's mother id and trait

FKM *KM for continuous Traits in Familial GWAS Data (calculate p-value)*

Description

This function (FKM) is used to perform famSKAT analysis (Chen et al., 2013) for continuous traits in familial GWAS data

Usage

```
FKM(obj, genotypes, gid, weights = NULL, acc = 1e-04, append.write = NULL)
```

Arguments

obj	results saved from FKM_Null_Model.R.
genotypes	1st column: gene name; 2nd column: snp name; 3rd-end columns: A matrix of genotypes for each subject (class: data.frame). The order of 3rd-end columns should match id. Coded as 0, 1, 2 and no missing. This genotype file can be a big file containing all genes or it can be files containing one single gene.
gid	A vector of id mapping to samples in genotype file (class: vector). So the order of samples in gid must be the same as the order in genotypes. Make sure it is not a factor. Although gid doesn't have to be in the same order as id, it is suggested to make them sorted in the same order in order to make all files easily to be tracked. No missing.

weights	1st column: gene name; 2nd column: snp name; 3rd column: A vector with the length equal to the number of variants in the test (class: data.frame). Default is Null indicating equal weight for all markers
acc	Accuracy of numerical integration used in Davies' method. Default 1e-4.
append.write	The name of pvalue output file. Write out p-values in real time. Don't need to wait until all genes are processed to get the final output.

Value

output: continuous trait family KM (F-KM) p-value

Examples

```
#########################################################################
### Examples for Continuous Traits in Familial GWAS Data using KM ###
#########################################################################
### Subject IDs are numeric ###
data("FKM_numID")
obj1 <- FKM_Null_Model(phenotype=fkm_n_y$y, id=fkm_n_y$id, fa=fkm_n_y$fa,
mo=fkm_n_y$mo, covariates=NULL)
pvalue1 <- FKM(obj=obj1, genotypes=fkm_n_gene, gid=fkm_n_geneid$gid, weights=NULL,
append.write="./pvalues.out")
# Read in a list of genes files instead of a big file containing all genes
obj <- FKM_Null_Model(phenotype=fkm_n_y$y, id=fkm_n_y$id, fa=fkm_n_y$fa, mo=fkm_n_y$mo,
covariates=NULL)
gene <- split(fkm_n_gene, fkm_n_gene[,1])
for (k in 1:2) {
  gene[[k]]$gene <- as.character(gene[[k]]$gene)
  pvalue1 <- FKM(obj=obj, genotypes=gene[[k]], gid=fkm_n_geneid$gid, weights=NULL,
  append.write="./pvalues.out")
}
### Subject IDs are character ###
data("FKM_charID")
obj1 <- FKM_Null_Model(phenotype=fkm_c_y$y, id=as.character(fkm_c_y$id),
fa=as.character(fkm_c_y$fa), mo=as.character(fkm_c_y$mo),  covariates=NULL)
pvalue1 <- FKM(obj=obj1, genotypes=fkm_c_gene, gid=as.character(fkm_c_geneid$gid),
weights=NULL)
```

FKMO *Optimal KM for continuous Traits in Familial GWAS Data (calculate p-value)*

Description

This function (FKMO) is used to perform optimal famSKAT analysis for continuous traits in familial GWAS data

Usage

```
FKMO(obj, genotypes, gid, weights = NULL, acc = 1e-04, acc2 = 1e-04,
  r.all = c(0, 0.25, 0.5, 0.75, 1), append.write = NULL)
```

Arguments

obj	results saved from FKMO_Null_Model.R.
genotypes	1st column: gene name; 2nd column: snp name; 3rd-end columns: A matrix of genotypes for each subject (class: data.frame). The order of 3rd-end columns should match id. Coded as 0, 1, 2 and no missing. This genotype file can be a big file containing all genes or it can be files containing one single gene.
gid	A vector of id mapping to samples in genotype file (class: vector). So the order of samples in gid must be the same as the order in genotypes. Make sure it is not a factor. Although gid doesn't have to be in the same order as id, it is suggested to make them sorted in the same order in order to make all files easily to be tracked. No missing.
weights	1st column: gene name; 2nd column: snp name; 3rd column: A vector with the length equal to the number of variants in the test (class: data.frame). Default is Null indicating equal weight for all markers
acc	Accuracy of numerical integration used in Davies' method for individual r.all p-values. Default 1e-4.
acc2	Accuracy of numerical integration used in Davies' method for the final p-value. Default 1e-4.
r.all	A list of predefined proportion of linear kernel and burden test. When r.all=0, regular kernel machine test (FKM); when r.all=1, burden test.
append.write	The name of pvalue output file. Write out p-values in real time. Don't need to wait until all genes are processed to get the final output.

Value

output: continuous trait optimal family KM (F-KMO) p-value

Examples

```
##############################################################################
### Examples for Continuous Traits in Familial GWAS Data using optimal KM ###
##############################################################################
### Subject IDs are numeric ###
data("FKM_numID")
obj1 <- FKMO_Null_Model(phenotype=fkm_n_y$y, id=fkm_n_y$id, fa=fkm_n_y$fa,
mo=fkm_n_y$mo, covariates=NULL)
pvalue1 <- FKMO(obj=obj1, genotypes=fkm_n_gene, gid=fkm_n_geneid$gid, weights=NULL,
append.write="./pvalues.out")
# Read in a list of genes files instead of a big file containing all genes
obj <- FKMO_Null_Model(phenotype=fkm_n_y$y, id=fkm_n_y$id, fa=fkm_n_y$fa,
mo=fkm_n_y$mo, covariates=NULL)
gene <- split(fkm_n_gene, fkm_n_gene[,1])
for (k in 1:2) {
```

```
  gene[[k]]$gene <- as.character(gene[[k]]$gene)
  pvalue1 <- FKMO(obj=obj, genotypes=gene[[k]], gid=fkm_n_geneid$gid, weights=NULL,
  append.write="./pvalues.out")
}
### Subject IDs are character ###
data("FKM_charID")
obj1 <- FKMO_Null_Model(phenotype=fkm_c_y$y, id=as.character(fkm_c_y$id),
fa=as.character(fkm_c_y$fa), mo=as.character(fkm_c_y$mo),  covariates=NULL)
pvalue1 <- FKMO(obj=obj1, genotypes=fkm_c_gene, gid=as.character(fkm_c_geneid$gid),
weights=NULL)
```

`FKMO_Null_Model` *Optimal KM for continuous Traits in Familial GWAS Data (fit null model)*

Description

This function (FKMO) is used to perform optimal famSKAT analysis for continuous traits in familial GWAS data
In the final correlation matrix, the covariance between parent and offspring is 0.5, the covariance between siblings is also 0.5

Usage

```
FKMO_Null_Model(phenotype, id, fa, mo, covariates = NULL)
```

Arguments

`phenotype`	A vector of quantitative trait in the analysis (class: vector). The order should match the vector id. Subjects with missing phenotypes are only used for kinship calculation.
`id`	A vector of id (class: vector). It can be either numeric or character. The id indicates each subject. Make sure it is not factor. No missing.
`fa`	A vector of father id (class: vector). It can be either numeric or character. The father id indicates the father of each subject. If this subject has no father in this data, the value is set to "NA". Make sure it is not factor.
`mo`	A vector of mother id (class: vector). It can be either numeric or character. The mother id indicates the mother of each subject. If this subject has no mother in this data, the value is set to "NA". Make sure it is not factor.
`covariates`	A matrix of covariates (class: data.frame). The order of rows should match the vector id. Default NULL. Subjects with missing covariates are only used for kinship calculation.

Value

output: object as input for FKMO

FKM_charID *This is continuous family data with character IDs*

Description

data.frame

Usage

```
data(FKM_charID)
```

Details

- fkm_c_covariates. age and sex for 100 samples
- fkm_c_gene. genotypes for 100 samples for two genes and each gene has 50 SNPs
- fkm_c_geneid. ordered sample id as in fkm_c_gene (100 samples)
- fkm_c_weights. weight for each SNP
- fkm_c_y. sample id, sample's father id, sample's mother id and trait

FKM_Null_Model *KM for continuous Traits in Familial GWAS Data (fit null model)*

Description

This function (FKM) is used to perform famSKAT analysis (Chen et al., 2013) for continuous traits in familial GWAS data
In the final correlation matrix, the covariance between parent and offspring is 0.5, the covariance between siblings is also 0.5

Usage

```
FKM_Null_Model(phenotype, id, fa, mo, covariates = NULL)
```

Arguments

phenotype	A vector of quantitative trait in the analysis (class: vector). The order should match the vector id. Subjects with missing phenotypes are only used for kinship calculation.
id	A vector of id (class: vector). It can be either numeric or character. The id indicates each subject. Make sure it is not factor. No missing.
fa	A vector of father id (class: vector). It can be either numeric or character. The father id indicates the father of each subject. If this subject has no father in this data, the value is set to "NA". Make sure it is not factor.

mo	A vector of mother id (class: vector). It can be either numeric or character. The mother id indicates the mother of each subject. If this subject has no mother in this data, the value is set to "NA". Make sure it is not factor.
covariates	A matrix of covariates (class: data.frame). The order of rows should match the vector id. Default NULL. Subjects with missing covariates are only used for kinship calculation.

Value

output: object as input for FKM

FKM_numID *This is continuous family data numeric IDs*

Description

data.frame

Usage

```
data(FKM_numID)
```

Details

- fkm_n_covariates. age and sex for 100 samples
- fkm_n_gene. genotypes for 100 samples for two genes and each gene has 50 SNPs
- fkm_n_geneid. ordered sample id as in fkm_n_gene (100 samples)
- fkm_n_weights. weight for each SNP
- fkm_n_y. sample id, sample's father id, sample's mother id and trait

---

LKM *KM for Quantitative Traits in Longitudinal GWAS Data (calculate p-value)*

Description

This function (LKM) is used to perform Kernel Machine analysis (Yan et al., 2016) for quantitative traits in GWAS longitudinal data.

Usage

```
LKM(obj, genotypes, gid, weights = NULL, acc = 1e-04, append.write = NULL)
```

Arguments

obj	results saved from LKM_Null_Model
genotypes	1st column: gene name; 2nd column: snp name; 3rd-end columns: A matrix of genotypes for each subject (class: data.frame). The order of 3rd-end columns should match unique(yid). Coded as 0, 1, 2 and no missing. This genotype file can be a big file containing all genes or it can be files containing one single gene.
gid	A vector of id mapping to samples in genotype file (class: vector). So the order of samples in gid must be the same as the order in genotypes. Make sure it is not a factor. Although gid doesn't have to be in the same order as yid, it is suggested to make them sorted in the same order in order to make all files easily to be tracked. No missing.
weights	1st column: gene name; 2nd column: snp name; 3rd column: A vector with the length equal to the number of variants in the test (class: data.frame). Default is Null indicating equal weight for all markers
acc	Accuracy of numerical integration used in Davies' method. Default 1e-4.
append.write	The name of pvalue output file. Write out p-values in real time. Don't need to wait until all genes are processed to get the final output.

Value

output: longitudinal KM (L-KM) p-value

Examples

```
#########################################################################
### Examples for Longitudinal Continuous Traits in GWAS Data using KM ###
#########################################################################
### Subject IDs are numeric ###
data("LKM_numID")
obj1 <- LKM_Null_Model(phenotype=lkm_n_y$y, time=lkm_n_y$time, yid=lkm_n_y$id,
covariates=NULL)
pvalue1 <- LKM(obj=obj1, genotypes=lkm_n_gene, gid=lkm_n_gid$gid, weights=NULL,
append.write="./pvalues.out")
# Read in a list of genes files instead of a big file containing all genes
obj <- LKM_Null_Model(phenotype=lkm_n_y$y, time=lkm_n_y$time, yid=lkm_n_y$id,
covariates=NULL)
gene <- split(lkm_n_gene, lkm_n_gene[,1])
for (k in 1:2) {
  gene[[k]]$gene <- as.character(gene[[k]]$gene)
  pvalue1 <- LKM(obj=obj, genotypes=gene[[k]], gid=lkm_n_gid$gid, weights=NULL,
  append.write="./pvalues.out")
}
### Subject IDs are character ###
data("LKM_charID")
obj1 <- LKM_Null_Model(phenotype=lkm_c_y$y, time=lkm_c_y$time,
yid=as.character(lkm_c_y$id), covariates=NULL)
pvalue1 <- LKM(obj=obj1, genotypes=lkm_c_gene, gid=as.character(lkm_c_gid$gid),
weights=NULL)
```

LKMO *Optimal KM for Quantitative Traits in Longitudinal GWAS Data (calculate p-value)*

Description

This function (LKMO) is used to perform optimal KM analysis for quantitative traits in GWAS longitudinal data.

Usage

```
LKMO(obj, genotypes, gid, weights = NULL, acc = 1e-04, acc2 = 1e-04,
  r.all = c(0, 0.25, 0.5, 0.75, 1), append.write = NULL)
```

Arguments

obj	results saved from LKMO_Null_Model
genotypes	1st column: gene name; 2nd column: snp name; 3rd-end columns: A matrix of genotypes for each subject (class: data.frame). The order of 3rd-end columns should match unique(yid). Coded as 0, 1, 2 and no missing. This genotype file can be a big file containing all genes or it can be files containing one single gene.
gid	A vector of id mapping to samples in genotype file (class: vector). So the order of samples in gid must be the same as the order in genotypes. Make sure it is not a factor. Although gid doesn't have to be in the same order as yid, it is suggested to make them sorted in the same order in order to make all files easily to be tracked. No missing.
weights	1st column: gene name; 2nd column: snp name; 3rd column: A vector with the length equal to the number of variants in the test (class: data.frame). Default is Null indicating equal weight for all markers
acc	Accuracy of numerical integration used in Davies' method for individual r.all p-values. Default 1e-4.
acc2	Accuracy of numerical integration used in Davies' method for the final p-value. Default 1e-4.
r.all	A list of predefined proportion of linear kernel and burden test. When r.all=0, regular kernel machine test (LKM); when r.all=1, burden test.
append.write	The name of pvalue output file. Write out p-values in real time. Don't need to wait until all genes are processed to get the final output.

Value

output: optimal longitudinal KM (L-KMO) p-value

Examples

```
##############################################################################
### Examples for Longitudinal Continuous Traits in GWAS Data using optimal KM ###
##############################################################################
### Subject IDs are numeric ###
data("LKM_numID")
obj1 <- LKMO_Null_Model(phenotype=lkm_n_y$y, time=lkm_n_y$time, yid=lkm_n_y$id,
covariates=NULL)
pvalue1 <- LKMO(obj=obj1, genotypes=lkm_n_gene, gid=lkm_n_gid$gid, weights=NULL,
append.write="./pvalues.out")
# Read in a list of genes files instead of a big file containing all genes
obj <- LKMO_Null_Model(phenotype=lkm_n_y$y, time=lkm_n_y$time, yid=lkm_n_y$id,
covariates=NULL)
gene <- split(lkm_n_gene, lkm_n_gene[,1])
for (k in 1:2) {
  gene[[k]]$gene <- as.character(gene[[k]]$gene)
  pvalue1 <- LKMO(obj=obj, genotypes=gene[[k]], gid=lkm_n_gid$gid, weights=NULL,
  append.write="./pvalues.out")
}
### Subject IDs are character ###
data("LKM_charID")
obj1 <- LKMO_Null_Model(phenotype=lkm_c_y$y, time=lkm_c_y$time,
yid=as.character(lkm_c_y$id), covariates=NULL)
pvalue1 <- LKMO(obj=obj1, genotypes=lkm_c_gene, gid=as.character(lkm_c_gid$gid),
weights=NULL)
```

`LKMO_Null_Model` *Optimal KM for Quantitative Traits in Longitudinal GWAS Data (fit null model)*

Description

This function (LKMO) is used to perform optimal KM analysis (Yan et al., 2016) for quantitative traits in GWAS longitudinal data.
It considers random intercept and random time

Usage

```
LKMO_Null_Model(phenotype, time, yid, covariates = NULL)
```

Arguments

`phenotype`	A vector of quantitative trait in the analysis (class: vector). The order should match the vector yid. No missing.
`time`	A vector of time points (class: vector). The order should match the vector yid. No missing.

yid	A vector of id (class: vector). Although it doesn't have to be sorted, observations from the same subject have to be connected with each other. The repeated id numbers indicate multiple time points for one subject. Make sure it is not a factor. No missing.
covariates	A matrix of covariates (class: data.frame). The order of rows should match the vector yid. Default NULL. No missing.

Value

output: object as input for LKMO

LKM_charID *This is continuous longitudinal data with character IDs*

Description

data.frame

Usage

```
data(LKM_charID)
```

Details

- lkm_c_covariates. age and sex for 20 samples, each one has 5 time points
- lkm_c_gene. genotypes for 20 samples for two genes and each gene has 50 SNPs
- lkm_c_gid. ordered sample id as in lkm_c_gene (20 samples)
- lkm_c_weights. weight for each SNP
- lkm_c_y. sample id, trait and time points

LKM_Int *KM for Traits by Time Interaction in Longitudinal GWAS Data*

Description

This function (LKM_Int) is used to perform Kernel Machine analysis for quantitative traits by time interaction in GWAS longitudinal data.
It considers random intercept and random time

Usage

```
LKM_Int(phenotype, time, yid, genotypes, gid, covariates = NULL,
  acc = 1e-04, append.write = NULL)
```

Arguments

phenotype	A vector of quantitative trait in the analysis (class: vector). The order should match the vector yid. No missing.
time	A vector of time points (class: vector). The order should match the vector yid. No missing.
yid	A vector of id (class: vector). Although it doesn't have to be sorted, observations from the same subject have to be connected with each other. The repeated id numbers indicate multiple time points for one subject. Make sure it is not a factor. No missing.
genotypes	1st column: gene name; 2nd column: snp name; 3rd-end columns: A matrix of genotypes for each subject (class: data.frame). The order of 3rd-end columns should match unique(yid). Coded as 0, 1, 2 and no missing. This genotype file can be a big file containing all genes or it can be files containing one single gene.
gid	A vector of id mapping to samples in genotype file (class: vector). So the order of samples in gid must be the same as the order in genotypes. Make sure it is not a factor. Although gid doesn't have to be in the same order as yid, it is suggested to make them sorted in the same order in order to make all files easily to be tracked. No missing.
covariates	A matrix of covariates (class: data.frame). The order of rows should match the vector yid. Default NULL. No missing.
acc	Accuracy of numerical integration used in Davies' method. Default 1e-4.
append.write	The name of pvalue output file. Write out p-values in real time. Don't need to wait until all genes are processed to get the final output.

Value

output: longitudinal time by marker interaction (LKM_Int) p-value

Examples

```
##############################################################################################
### Examples for Marker by Time Interaction in Longitudinal Continuous Traits in GWAS
##############################################################################################
###################
# Data using KM ###
###################
data("LKM_numID_int")
pvalue1 <- LKM_Int(phenotype=lkm_int_n_y$y, genotypes=lkm_int_n_gene, time=lkm_int_n_y$time,
yid=lkm_int_n_y$id, gid=lkm_int_n_gid$gid, covariates=NULL, append.write="./pvalues.out")
```

LKM_Null_Model *KM for Quantitative Traits in Longitudinal GWAS Data (fit null model)*

Description

This function (LKM) is used to perform Kernel Machine analysis (Yan et al., 2016) for quantitative traits in GWAS longitudinal data.
It considers random intercept and random time

Usage

```
LKM_Null_Model(phenotype, time, yid, covariates = NULL)
```

Arguments

phenotype	A vector of quantitative trait in the analysis (class: vector). The order should match the vector yid. No missing.
time	A vector of time points (class: vector). The order should match the vector yid. No missing.
yid	A vector of id (class: vector). Although it doesn't have to be sorted, observations from the same subject have to be connected with each other. The repeated id numbers indicate multiple time points for one subject. Make sure it is not a factor. No missing.
covariates	A matrix of covariates (class: data.frame). The order of rows should match the vector yid. Default NULL. No missing.

Value

output: object as input for LKM

LKM_numID *This is continuous longitudinal data with numeric IDs*

Description

data.frame

Usage

```
data(LKM_numID)
```

Details

- lkm_n_covariates. age and sex for 20 samples, each one has 5 time points
- lkm_n_gene. genotypes for 20 samples for two genes and each gene has 50 SNPs
- lkm_n_gid. ordered sample id as in lkm_n_gene (20 samples)
- lkm_n_weights. weight for each SNP
- lkm_n_y. sample id, trait and time points

`LKM_numID_int` *This is continuous longitudinal data with numeric IDs*

Description

data.frame

Usage

```
data(LKM_numID_int)
```

Details

- lkm_int_n_covariates. age and sex for 200 samples, each one has 5 time points
- lkm_int_n_gene. genotypes for 200 samples for two genes and each gene has 50 SNPs
- lkm_int_n_gid. ordered sample id as in lkm_int_n_gene (200 samples)
- lkm_int_n_weights. weight for each SNP
- lkm_int_n_y. sample id, trait and time points

`MFKM` *KM for Quantitative Traits in Multivariate Family GWAS Data (calculate p-value)*

Description

This function (MFKM) is used to perform KM analysis (Yan et al., 2015) for quantitative traits in GWAS multivariate family data.

Usage

```
MFKM(obj, genotypes, weights = NULL, acc = 1e-04, append.write = NULL,
  eq.gen.effect = F)
```

Arguments

obj	results saved from MFKM_Null_Model.
genotypes	1st column: gene name; 2nd column: snp name; 3rd-end columns: A matrix of genotypes for each subject (class: data.frame). The order of 3rd-end columns should match unique(yid). Coded as 0, 1, 2 and no missing. This genotype file can be a big file containing all genes or it can be files containing one single gene.
weights	1st column: gene name; 2nd column: snp name; 3rd column: A vector with the length equal to the number of variants in the test (class: data.frame). Default is Null indicating equal weight for all markers.
acc	Accuracy of numerical integration used in Davies' method. Default 1e-4.
append.write	The name of pvalue output file. Write out p-values in real time. Don't need to wait until all genes are processed to get the final output.
eq.gen.effect	Whether assume equal genetic effects on different traits (default = False).

Value

output: multivariate family KM (MF-KM) p-value

Examples

```
##############################################################################################
### Examples for Multivariate (two) Continuous Traits in Familial GWAS Data using KM ###
##############################################################################################
### Subject IDs are numeric ###
data("MFKM_numID")
#If Ninitial=1, the initial value "cor" is always equal to
#correlation(trait1|covariates, trait2|covariates).
obj1 <- MFKM_Null_Model(phenotype=mfkm_n_y$y, trait=mfkm_n_y$trait, yid=mfkm_n_y$id,
gid=mfkm_n_geneid$gid, fa=mfkm_n_geneid$fa, mo=mfkm_n_geneid$mo, covariates=NULL,
Ninitial=1)
#One should try multiple initial values in order to find max log-likelihood. The default
#is 10 times.
#This could be time consuming, depends on the sample size. The good thing is that null
#model only needs to be fitted once for the whole genome, so it's worth trying many
#initial values. The initial value with max logl can be saved in ./LogLikelihood.txt
#for reuse.
pvalue1 <- MFKM(obj=obj1, genotypes=mfkm_n_gene, weights=NULL)
#If one wants to replicate the results, finds the initial value "cor" with the max
#loglikelihood in ./LogLikelihood.txt that is output by default. The 1st column is
#"cor"; 2nd column is "log-likelihood".
obj1 <- MFKM_Null_Model(phenotype=mfkm_n_y$y, trait=mfkm_n_y$trait, yid=mfkm_n_y$id,
gid=mfkm_n_geneid$gid, fa=mfkm_n_geneid$fa, mo=mfkm_n_geneid$mo, covariates=NULL,
cor=0.687439771651474)
#This "cor" is calcualted based on 3 initial values, "Ninitial=3".
pvalue1 <- MFKM(obj=obj1, genotypes=mfkm_n_gene, weights=NULL)
#Introduce missing into covariates and outcome.
#When samples have missing values in outcome or covariates, those samples are used only for
#kinship calculation.
mfkm_n_covariates[1,] = NA
```

```
mfkm_n_y$y[4] = NA
obj1 <- MFKM_Null_Model(phenotype=mfkm_n_y$y, trait=mfkm_n_y$trait, yid=mfkm_n_y$id,
gid=mfkm_n_geneid$gid, fa=mfkm_n_geneid$fa, mo=mfkm_n_geneid$mo, covariates=mfkm_n_covariates,
Ninitial=1)
pvalue1 <- MFKM(obj=obj1, genotypes=mfkm_n_gene, weights=NULL)
#Read in a list of genes files instead of a big file containing all genes
obj <- MFKM_Null_Model(phenotype=mfkm_n_y$y, trait=mfkm_n_y$trait, yid=mfkm_n_y$id,
gid=mfkm_n_geneid$gid, fa=mfkm_n_geneid$fa, mo=mfkm_n_geneid$mo, covariates=NULL,
Ninitial=1)
gene <- split(mfkm_n_gene, mfkm_n_gene[,1])
for (k in 1:2) {
  gene[[k]]$gene <- as.character(gene[[k]]$gene)
  pvalue1 <- MFKM(obj=obj, genotypes=gene[[k]], weights=NULL, append.write=
  "./pvalues.out")
}
### Subject IDs are character ###
data("MFKM_charID")
obj1 <- MFKM_Null_Model(phenotype=mfkm_c_y$y, trait=mfkm_c_y$trait,
yid=as.character(mfkm_c_y$id),
gid=as.character(mfkm_c_geneid$gid), fa=as.character(mfkm_c_geneid$fa),
mo=as.character(mfkm_c_geneid$mo), covariates=NULL, Ninitial=1)
pvalue1 <- MFKM(obj=obj1, genotypes=mfkm_c_gene, weights=NULL)
```

MFKMO *Optimal KM for Quantitative Traits in Multivariate Family GWAS Data (calculate p-value)*

Description

This function (MFKMO) is used to perform optimal KM analysis for quantitative traits in GWAS multivariate family data.

Usage

```
MFKMO(obj, genotypes, weights = NULL, acc = 1e-04, acc2 = 1e-04,
  r.all = c(0, 0.25, 0.5, 0.75, 1), append.write = NULL,
  eq.gen.effect = F)
```

Arguments

obj	results saved from MFKMO_Null_Model.
genotypes	1st column: gene name; 2nd column: snp name; 3rd-end columns: A matrix of genotypes for each subject (class: data.frame). The order of 3rd-end columns should match unique(yid). Coded as 0, 1, 2 and no missing. This genotype file can be a big file containing all genes or it can be files containing one single gene.
weights	1st column: gene name; 2nd column: snp name; 3rd column: A vector with the length equal to the number of variants in the test (class: data.frame). Default is Null indicating equal weight for all markers

acc	Accuracy of numerical integration used in Davies' method for individual r.all p-values. Default 1e-4.
acc2	Accuracy of numerical integration used in Davies' method for the final p-value. Default 1e-4.
r.all	A list of predefined proportion of linear kernel and burden test. When r.all=0, regular kernel machine test (MFKM); when r.all=1, burden test.
append.write	The name of pvalue output file. Write out p-values in real time. Don't need to wait until all genes are processed to get the final output.
eq.gen.effect	Whether assume equal genetic effects on different traits (default = False).

Value

output: optimal multivariate family KM (MF-KMO) p-value

Examples

```
##############################################################################
### Examples for Multivariate (two) Continuous Traits in Familial GWAS Data
##############################################################################
######################
# using optimal KM ###
######################
### Subject IDs are numeric ###
data("MFKM_numID")
# Read in a list of genes files instead of a big file containing all genes
obj <- MFKMO_Null_Model(phenotype=mfkm_n_y$y, trait=mfkm_n_y$trait, yid=mfkm_n_y$id,
gid=mfkm_n_geneid$gid, fa=mfkm_n_geneid$fa, mo=mfkm_n_geneid$mo, covariates=NULL,
Ninitial=1)
gene <- split(mfkm_n_gene, mfkm_n_gene[,1])
for (k in 1:2) {
  gene[[k]]$gene <- as.character(gene[[k]]$gene)
  pvalue1 <- MFKMO(obj=obj, genotypes=gene[[k]], weights=NULL, append.write=
  "./pvalues.out")
}
### Subject IDs are character ###
data("MFKM_charID")
obj1 <- MFKMO_Null_Model(phenotype=mfkm_c_y$y, trait=mfkm_c_y$trait,
yid=as.character(mfkm_c_y$id),
gid=as.character(mfkm_c_geneid$gid), fa=as.character(mfkm_c_geneid$fa),
mo=as.character(mfkm_c_geneid$mo), covariates=NULL, Ninitial=1)
pvalue1 <- MFKMO(obj=obj1, genotypes=mfkm_c_gene, weights=NULL)
```

MFKMO_Null_Model *Optimal KM for Quantitative Traits in Multivariate Family GWAS Data (fit null model)*

Description

This function (MFKMO) is used to perform optimal KM analysis for quantitative traits in GWAS multivariate family data.
It takes familial correlation as a kinship matrix

Usage

```
MFKMO_Null_Model(phenotype, trait, yid, gid, fa, mo, covariates = NULL,
  Ninitial = 10, method = "Nelder-Mead",
  LL.output = "./LogLikelihood.txt", cor = NULL, eq.cov.effect = F)
```

Arguments

phenotype	A vector of quantitative trait in the analysis (class: vector). The order should match the vector yid. Subjects with missing phenotypes are only used for kinship calculation.
trait	A vector of multivariate traits (class: vector). The order should match the vector yid. No missing.
yid	A vector of id (class: vector). Although it doesn't have to be sorted, observations from the same subject have to be connected with each other. The repeated id numbers indicate multiple time points for one subject. Make sure it is not a factor. No missing.
gid	A vector of id mapping to samples in genotype file (class: vector). So the order of samples in gid must be the same as the order in genotypes. Make sure it is not a factor. Although gid doesn't have to be in the same order as yid, it is suggested to make them sorted in the same order in order to make all files easily to be tracked. No missing.
fa	A vector of father id (class: vector). The father id indicates the father of each subject. If this subject has no father in this data, the value is set to "NA". Make sure it is not factor.
mo	A vector of mother id (class: vector). The mother id indicates the mother of each subject. If this subject has no mother in this data, the value is set to "NA". Make sure it is not factor.
covariates	A matrix of covariates (class: data.frame). The order of rows should match the vector yid. Default NULL. Subjects with missing covariates are only used for kinship calculation.
Ninitial	The number of times to try initial values. The default is 10 times. If Ninitial=1, the initial value "cor" is always equal to correlation(trait1|covariates, trait2|covariates). One should try multiple initial values in order to find max log-likelihood. This could be time consuming, depends on the sample size. The good thing is that null model only needs to be fitted once for the whole genome, so it's worth trying many initial values.
method	The optimization method used in null model. The default method is an implementation of that of Nelder and Mead (1965), that uses only function values and is robust but relatively slow. Method "L-BFGS-B" is that of Byrd et. al. (1995) which allows box constraints, that is each variable can be given a lower and/or upper bound.

LL.output	Output all tried initial values and corresponding log-likelihoods. The initial value with max log-likelihood is used in the algorithm and it can be used for replication. The output file can be renamed.
cor	Initial value. By default, it's not given, the program tries to find the best initial value. Once it's given, the program uses it as the only initial value. This is useful when one already knows the initial value corresponding to max log-likelihood.
eq.cov.effect	Whether assume equal covariates effects on different traits (Default=False).

Value

output: object as input for MFKMO

MFKM_charID *This is continuous multivariate family data with character IDs*

Description

data.frame

Usage

```
data(MFKM_charID)
```

Details

- mfkm_c_covariates. age, sex and pc(trait specific covariate) for 50 samples, each one has two traits
- mfkm_c_gene. genotypes for 50 samples for two genes and each gene has 50 SNPs
- mfkm_c_geneid. ordered sample id as in mfkm_c_gene (50 samples)
- mfkm_c_weights. weight for each SNP
- mfkm_c_y. sample id, trait value and trait label

MFKM_Null_Model *KM for Quantitative Traits in Multivariate Family GWAS Data (fit null model)*

Description

This function (MFKM) is used to perform KM analysis (Yan et al., 2015) for quantitative traits in GWAS multivariate family data.
It takes familial correlation as a kinship matrix

Usage

```
MFKM_Null_Model(phenotype, trait, yid, gid, fa, mo, covariates = NULL,
  Ninitial = 10, method = "Nelder-Mead",
  LL.output = "./LogLikelihood.txt", cor = NULL, eq.cov.effect = F)
```

Arguments

phenotype	A vector of quantitative trait in the analysis (class: vector). The order should match the vector yid. Subjects with missing phenotypes are only used for kinship calculation.
trait	A vector of multivariate traits (class: vector). The order should match the vector yid. No missing.
yid	A vector of id (class: vector). Although it doesn't have to be sorted, observations from the same subject have to be connected with each other. The repeated id numbers indicate mutiple time points for one subject. Make sure it is not a factor. No missing.
gid	A vector of id mapping to samples in genotype file (class: vector). So the order of samples in gid must be the same as the order in genotypes. Make sure it is not a factor. Although gid doesn't have to be in the same order as yid, it is suggested to make them sorted in the same order in order to make all files easily to be tracked. No missing.
fa	A vector of father id (class: vector). The father id indicates the father of each subject. If this subject has no father in this data, the value is set to "NA". Make sure it is not factor.
mo	A vector of mother id (class: vector). The mother id indicates the mother of each subject. If this subject has no mother in this data, the value is set to "NA". Make sure it is not factor.
covariates	A matrix of covariates (class: data.frame). The order of rows should match the vector yid. Default NULL. Subjects with missing covariates are only used for kinship calculation.
Ninitial	The number of times to try initial values. The default is 10 times. If Ninitial=1, the initial value "cor" is always equal to correlation(trait1|covariates, trait2|covariates). One should try multiple initial values in order to find max log-likelihood. This could be time consuming, depends on the sample size. The good thing is that null model only needs to be fitted once for the whole genome, so it's worth trying many initial values.
method	The optimization method used in null model. The default method is an implementation of that of Nelder and Mead (1965), that uses only function values and is robust but relatively slow. Method "L-BFGS-B" is that of Byrd et. al. (1995) which allows box constraints, that is each variable can be given a lower and/or upper bound.
LL.output	Output all tried initial values and corresponding log-likelihoods. The initial value with max log-likelihood is used in the algorithm and it can be used for replication. The output file can be renamed.

cor	Initial value. By default, it's not given, the program tries to find the best initial value. Once it's given, the program uses it as the only initial value. This is useful when one already knows the initial value corresponding to max log-likelihood.
eq.cov.effect	Whether assume equal covariates effects on different traits (Default=False).

Value

output: object as input for MFKM

`MFKM_numID` *This is continuous multivariate family data with numeric IDs*

Description

data.frame

Usage

```
data(MFKM_numID)
```

Details

- mfkm_n_covariates. age, sex and pc(trait specific covariate) for 50 samples, each one has two traits
- mfkm_n_gene. genotypes for 50 samples for two genes and each gene has 50 SNPs
- mfkm_n_geneid. ordered sample id as in mfkm_n_gene (50 samples)
- mfkm_n_weights. weight for each SNP
- mfkm_n_y. sample id, trait value and trait label

`MKM` *KM for Quantitative Traits in Multivariate GWAS Data (calculate p-value)*

Description

This function (MKM) is used to perform KM analysis (Tzeng et al. 2012) for quantitative traits in GWAS multivariate data.

Usage

```
MKM(obj, genotypes, gid, weights = NULL, acc = 1e-04, append.write = NULL,
  eq.gen.effect = F)
```

Arguments

obj	results saved from MKM_Null_Model.
genotypes	1st column: gene name; 2nd column: snp name; 3rd-end columns: A matrix of genotypes for each subject (class: data.frame). The order of 3rd-end columns should match unique(yid). Coded as 0, 1, 2 and no missing. This genotype file can be a big file containing all genes or it can be files containing one single gene.
gid	A vector of id mapping to samples in genotype file (class: vector). So the order of samples in gid must be the same as the order in genotypes. Make sure it is not a factor. Although gid doesn't have to be in the same order as yid, it is suggested to make them sorted in the same order in order to make all files easily to be tracked. No missing.
weights	1st column: gene name; 2nd column: snp name; 3rd column: A vector with the length equal to the number of variants in the test (class: data.frame). Default is Null indicating equal weight for all markers
acc	Accuracy of numerical integration used in Davies' method. Default 1e-4.
append.write	The name of pvalue output file. Write out p-values in real time. Don't need to wait until all genes are processed to get the final output.
eq.gen.effect	Whether assume equal genetic effects on different traits (default = False).

Value

output: multivariate KM (M-KM) p-value

Examples

```
##############################################################################
### Examples for Multivariate (two) Continuous Traits in GWAS Data using KM ###
##############################################################################
### Subject IDs are numeric ###
data("MKM_numID")
obj1 <- MKM_Null_Model(phenotype=mkm_n_ph$y, trait=mkm_n_ph$trait, yid=mkm_n_ph$id,
covariates=NULL)
pvalue1 <- MKM(obj=obj1, genotypes=mkm_n_gene, gid=mkm_n_geneid$gid, weights=NULL,
append.write="./pvalues.out")
# Read in a list of genes files instead of a big file containing all genes
obj <- MKM_Null_Model(phenotype=mkm_n_ph$y, trait=mkm_n_ph$trait, yid=mkm_n_ph$id,
covariates=NULL)
gene <- split(mkm_n_gene, mkm_n_gene[,1])
for (k in 1:2) {
  gene[[k]]$gene <- as.character(gene[[k]]$gene)
  pvalue1 <- MKM(obj=obj, genotypes=gene[[k]], gid=mkm_n_geneid$gid, weights=NULL,
  append.write="./pvalues.out")
}
### Subject IDs are character ###
data("MKM_charID")
obj1 <- MKM_Null_Model(phenotype=mkm_c_ph$y, trait=mkm_c_ph$trait,
yid=as.character(mkm_c_ph$id), covariates=NULL)
pvalue1 <- MKM(obj=obj1, genotypes=mkm_c_gene, gid=as.character(mkm_c_geneid$gid),
weights=NULL)
```

MKMO *Optimal KM for Quantitative Traits in Multivariate GWAS Data (calculate p-value)*

Description

This function (MKMO) is used to perform optimal KM analysis for quantitative traits in GWAS multivariate data.

Usage

```
MKMO(obj, genotypes, gid, weights = NULL, acc = 1e-04, acc2 = 1e-04,
  r.all = c(0, 0.25, 0.5, 0.75, 1), append.write = NULL,
  eq.gen.effect = F)
```

Arguments

obj	results saved from MKMO_Null_Model.
genotypes	1st column: gene name; 2nd column: snp name; 3rd-end columns: A matrix of genotypes for each subject (class: data.frame). The order of 3rd-end columns should match unique(yid). Coded as 0, 1, 2 and no missing. This genotype file can be a big file containing all genes or it can be files containing one single gene.
gid	A vector of id mapping to samples in genotype file (class: vector). So the order of samples in gid must be the same as the order in genotypes. Make sure it is not a factor. Although gid doesn't have to be in the same order as yid, it is suggested to make them sorted in the same order in order to make all files easily to be tracked. No missing.
weights	1st column: gene name; 2nd column: snp name; 3rd column: A vector with the length equal to the number of variants in the test (class: data.frame). Default is Null indicating equal weight for all markers
acc	Accuracy of numerical integration used in Davies' method for individual r.all p-values. Default 1e-4.
acc2	Accuracy of numerical integration used in Davies' method for the final p-value. Default 1e-4.
r.all	A list of predefined proportion of linear kernel and burden test. When r.all=0, regular kernel machine test (MKM); when r.all=1, burden test.
append.write	The name of pvalue output file. Write out p-values in real time. Don't need to wait until all genes are processed to get the final output.
eq.gen.effect	Whether assume equal genetic effects on different traits (default = False).

Value

output: optimal multivariate KM (M-KMO) p-value

Examples

```
##############################################################################################
### Examples for Multivariate (two) Continuous Traits in GWAS Data using optimal KM ###
##############################################################################################
### Subject IDs are numeric ###
data("MKM_numID")
obj1 <- MKMO_Null_Model(phenotype=mkm_n_ph$y, trait=mkm_n_ph$trait, yid=mkm_n_ph$id,
covariates=NULL)
pvalue1 <- MKMO(obj=obj1, genotypes=mkm_n_gene, gid=mkm_n_geneid$gid, weights=NULL,
append.write="./pvalues.out")
# Read in a list of genes files instead of a big file containing all genes
obj <- MKMO_Null_Model(phenotype=mkm_n_ph$y, trait=mkm_n_ph$trait, yid=mkm_n_ph$id,
covariates=NULL)
gene <- split(mkm_n_gene, mkm_n_gene[,1])
for (k in 1:2) {
  gene[[k]]$gene <- as.character(gene[[k]]$gene)
  pvalue1 <- MKMO(obj=obj, genotypes=gene[[k]], gid=mkm_n_geneid$gid, weights=NULL,
append.write="./pvalues.out")
}
### Subject IDs are character ###
data("MKM_charID")
obj1 <- MKMO_Null_Model(phenotype=mkm_c_ph$y, trait=mkm_c_ph$trait,
yid=as.character(mkm_c_ph$id), covariates=NULL)
pvalue1 <- MKMO(obj=obj1, genotypes=mkm_c_gene, gid=as.character(mkm_c_geneid$gid),
weights=NULL)
```

`MKMO_Null_Model` *Optimal KM for Multiple Quantitative Traits in GWAS Data (fit null model)*

Description

This function (MKMO) is used to perform optimal KM analysis for multiple (two) quantitative traits in GWAS data.
It considers variances of trait 1 and 2, and covariance between trait 1 and 2

Usage

```
MKMO_Null_Model(phenotype, trait, yid, covariates = NULL, eq.cov.effect = F)
```

Arguments

`phenotype`	A vector of quantitative trait in the analysis (class: vector). The order should match the vector yid. No missing.
`trait`	A vector of multivariate traits (class: vector). The order should match the vector yid. No missing.

yid	A vector of id (class: vector). Although it doesn't have to be sorted, observations from the same subject have to be connected with each other. The repeated id numbers indicate multiple traits for one subject. Make sure it is not a factor. No missing.
covariates	A matrix of covariates (class: data.frame). The order of rows should match the vector yid. Default NULL. No missing.
eq.cov.effect	Whether assume equal covariates effects on different traits (Default=False).

Value

output: object as input for MKMO

`MKM_charID` *This is continuous multivariate data with character IDs*

Description

data.frame

Usage

```
data(MKM_charID)
```

Details

- mkm_c_covariates. age and sex for 50 samples, each one has two traits
- mkm_c_gene. genotypes for 50 samples for two genes and each gene has 50 SNPs
- mkm_c_geneid. ordered sample id as in mkm_c_gene (50 samples)
- mkm_c_weights. weight for each SNP
- mkm_c_ph. sample id, trait value and trait label

`MKM_Null_Model` *KM for Multiple Quantitative Traits in GWAS Data (fit null model)*

Description

This function (MKM) is used to perform KM analysis (Tzeng et al. 2012) for multiple (two) quantitative traits in GWAS data.
It considers variances of trait 1 and 2, and covariance between trait 1 and 2

Usage

```
MKM_Null_Model(phenotype, trait, yid, covariates = NULL, eq.cov.effect = F)
```

Arguments

phenotype	A vector of quantitative trait in the analysis (class: vector). The order should match the vector yid. No missing.
trait	A vector of multivariate traits (class: vector). The order should match the vector yid. No missing.
yid	A vector of id (class: vector). Although it doesn't have to be sorted, observations from the same subject have to be connected with each other. The repeated id numbers indicate multiple traits for one subject. Make sure it is not a factor. No missing.
covariates	A matrix of covariates (class: data.frame). The order of rows should match the vector yid. Default NULL. No missing.
eq.cov.effect	Whether assume equal covariates effects on different traits (Default=False).

Value

output: object as input for MKM

MKM_numID *This is continuous multivariate data with numeric IDs*

Description

data.frame

Usage

```
data(MKM_numID)
```

Details

- mkm_n_covariates. age and sex for 50 samples, each one has two traits
- mkm_n_gene. genotypes for 50 samples for two genes and each gene has 50 SNPs
- mkm_n_geneid. ordered sample id as in mkm_n_gene (50 samples)
- mkm_n_weights. weight for each SNP
- mkm_n_ph. sample id, trait value and trait label

prepCoxKM *KM for Survival Traits in GWAS Data (fit null model)*

Description

This function (CoxKM) is used to perform KM analysis (Chen et al., 2014) for survival traits in GWAS data

Usage

```
prepCoxKM(Z, formula, SNPInfo = NULL, snpNames = "Name",
  aggregateBy = "gene", data = parent.frame(), verbose = FALSE)
```

Arguments

Z	A matrix of genotypes (class: matrix). rows correspond to individuals and columns correspond to SNPs. Use 'NA' for missing values. The column names of this matrix should correspond to SNP names in the SNP information file.
formula	Base formula under null hypothesis (class: formula). For Cox models, the formula follows that of the coxph() function.
SNPInfo	SNP Info file (class: data.frame). Must contain fields given in 'snpName' and 'aggregateBy'.
snpNames	The field of SNPInfo where the SNP identifiers are found. (Default="Name")
aggregateBy	The field of SNPInfo on which the skat results were aggregated. (Default="gene")
data	The data file in which to find variables in the formula (class: data.frame).
verbose	Whether or not to print the progress bar (class: logical).

Value

output: object as input for FbKM

Index