Package: qgg (via r-universe)

October 5, 2026

Type Package

Title Statistical Tools for Quantitative Genetic Analyses

Version 1.1.6

Date 2024-12-13

Maintainer Peter Soerensen <peter.sorensen@r-qgg.org>

Description Provides an infrastructure for efficient processing of large-scale genetic and phenotypic data including core functions for: 1) fitting linear mixed models, 2) constructing marker-based genomic relationship matrices, 3) estimating genetic parameters (heritability and correlation), 4) performing genomic prediction and genetic risk profiling, and 5) single or multi-marker association analyses. Rohde et al. (2019) <doi:10.1101/503631>.

License GPL-3

Encoding UTF-8

Depends R (>= 3.3.0)

Imports Rcpp (>= 1.0.5), data.table, parallel, statmod, stats, MCMCpack, MASS, coda, corpcor, Matrix, methods

LinkingTo Rcpp, RcppArmadillo

RoxygenNote 7.3.2

URL https://github.com/psoerensen/qgg

BugReports https://github.com/psoerensen/qgg/issues

NeedsCompilation yes

Author Peter Soerensen [aut, cre], Palle Duun Rohde [aut], Izel Fourie Soerensen [aut]

Repository https://cran.r-universe.dev

Date/Publication 2024-12-16 16:50:02 UTC

RemoteUrl https://github.com/cran/qgg

RemoteRef HEAD

RemoteSha 0f0aebc63cb30b482b65dd1151999aca814df3a2

Contents

acc . . . 2
adjStat . . . 3
gbayes . . . 4
getG . . . 8
gfilter . . . 9
glma . . . 10
gmap . . . 13
gprep . . . 16
greml . . . 18
grm . . . 21
gscore . . . 23
gsea . . . 25
gsim . . . 27
gsolve . . . 29
ldsc . . . 31
magma . . . 33
mtadj . . . 34
pops . . . 36
vegas . . . 38

Index **40**

`acc` *Compute prediction accuracy for a quantitative or binary trait*

Description

Compute prediction accuracy for a quantitative or binary trait

Usage

```
acc(yobs = NULL, ypred = NULL, typeoftrait = "quantitative")
```

Arguments

`yobs`	is a vector of observed phenotypes
`ypred`	is a vector of predicted phenotypes
`typeoftrait`	is a character with possible values "binary" or "quantitative" (default)

adjStat *Adjustment of marker summary statistics using clumping and thresholding*

Description

Adjust marker summary statistics using linkage disequilibrium information from Glist.

Usage

```
adjStat(
  stat = NULL,
  Glist = NULL,
  chr = NULL,
  statistics = "b",
  r2 = 0.9,
  ldSets = NULL,
  threshold = 1,
  header = NULL,
  method = "pruning"
)
```

Arguments

stat	A data frame with marker summary statistics (see required format above).
Glist	List of information about genotype matrix stored on disk.
chr	Chromosome(s) being processed.
statistics	Specify what type of statistics ("b" or "z") is being processed. Default is "b".
r2	Threshold used in clumping/pruning procedure. Default is 0.9.
ldSets	List of marker sets - names correspond to row names in 'stat'.
threshold	P-value threshold used in clumping procedure. Default is 1.
header	Character vector with column names to be excluded in the LD adjustment.
method	Method used in adjustment for linkage disequilibrium. Default is "clumping".

Details

Required input format for summary statistics:

stat can be a data.frame(rsids, chr, pos, ea, nea, eaf, b, seb, stat, p, n) (single trait)

stat can be a list(marker=(rsids, chr, pos, ea, nea, eaf), b, seb, stat, p, n) (multiple trait)

For details about the summary statistics format, see the main function description.

Author(s)

Peter Soerensen

gbayes *Bayesian linear regression models*

Description

Bayesian linear regression (BLR) models:

- unified mapping of genetic variants, estimation of genetic parameters (e.g. heritability) and prediction of disease risk)

- handles different genetic architectures (few large, many small effects)

- scale to large data (e.g. sparse LD)

In the Bayesian multiple regression model the posterior density of the model parameters depend on the likelihood of the data given the parameters and a prior probability for the model parameters

The prior density of marker effects defines whether the model will induce variable selection and shrinkage or shrinkage only. Also, the choice of prior will define the extent and type of shrinkage induced. Ideally the choice of prior for the marker effect should reflect the genetic architecture of the trait, and will vary (perhaps a lot) across traits.

The following prior distributions are provided:

Bayes N: Assigning a Gaussian prior to marker effects implies that the posterior means are the BLUP estimates (same as Ridge Regression).

Bayes L: Assigning a double-exponential or Laplace prior is the density used in the Bayesian LASSO

Bayes A: similar to ridge regression but t-distribution prior (rather than Gaussian) for the marker effects ; variance comes from an inverse-chi-square distribution instead of being fixed. Estimation via Gibbs sampling.

Bayes C: uses a "rounded spike" (low-variance Gaussian) at origin many small effects can contribute to polygenic component, reduces the dimensionality of the model (makes Gibbs sampling feasible).

Bayes R: Hierarchical Bayesian mixture model with 4 Gaussian components, with variances scaled by 0, 0.0001 , 0.001 , and 0.01 .

Usage

```
gbayes(
  y = NULL,
  X = NULL,
  W = NULL,
  stat = NULL,
  covs = NULL,
  trait = NULL,
  fit = NULL,
  Glist = NULL,
  chr = NULL,
  rsids = NULL,
```

```
  b = NULL,
  bm = NULL,
  seb = NULL,
  LD = NULL,
  n = NULL,
  formatLD = "dense",
  vg = NULL,
  vb = NULL,
  ve = NULL,
  ssg_prior = NULL,
  ssb_prior = NULL,
  sse_prior = NULL,
  lambda = NULL,
  scaleY = TRUE,
  h2 = NULL,
  pi = 0.001,
  updateB = TRUE,
  updateG = TRUE,
  updateE = TRUE,
  updatePi = TRUE,
  adjustE = TRUE,
  models = NULL,
  nug = 4,
  nub = 4,
  nue = 4,
  verbose = FALSE,
  msize = 100,
  mask = NULL,
  GRMlist = NULL,
  ve_prior = NULL,
  vg_prior = NULL,
  tol = 0.001,
  nit = 100,
  nburn = 0,
  nthin = 1,
  nit_local = NULL,
  nit_global = NULL,
  method = "mixed",
  algorithm = "mcmc"
)
```

Arguments

y	is a vector or matrix of phenotypes
X	is a matrix of covariates
W	is a matrix of centered and scaled genotypes
stat	dataframe with marker summary statistics
covs	is a list of summary statistics (output from internal cvs function)

trait	is an integer used for selection traits in covs object
fit	is a list of results from gbayes
Glist	list of information about genotype matrix stored on disk
chr	is the chromosome for which to fit BLR models
rsids	is a character vector of rsids
b	is a vector or matrix of marginal marker effects
bm	is a vector or matrix of adjusted marker effects for the BLR model
seb	is a vector or matrix of standard error of marginal effects
LD	is a list with sparse LD matrices
n	is a scalar or vector of number of observations for each trait
formatLD	is a character specifying LD format (formatLD="dense" is default)
vg	is a scalar or matrix of genetic (co)variances
vb	is a scalar or matrix of marker (co)variances
ve	is a scalar or matrix of residual (co)variances
ssg_prior	is a scalar or matrix of prior genetic (co)variances
ssb_prior	is a scalar or matrix of prior marker (co)variances
sse_prior	is a scalar or matrix of prior residual (co)variances
lambda	is a vector or matrix of lambda values
scaleY	is a logical; if TRUE y is centered and scaled
h2	is the trait heritability
pi	is the proportion of markers in each marker variance class (e.g. pi=c(0.999,0.001),used if method="ssvs")
updateB	is a logical for updating marker (co)variances
updateG	is a logical for updating genetic (co)variances
updateE	is a logical for updating residual (co)variances
updatePi	is a logical for updating pi
adjustE	is a logical for adjusting residual variance
models	is a list structure with models evaluated in bayesC
nug	is a scalar or vector of prior degrees of freedom for prior genetic (co)variances
nub	is a scalar or vector of prior degrees of freedom for marker (co)variances
nue	is a scalar or vector of prior degrees of freedom for prior residual (co)variances
verbose	is a logical; if TRUE it prints more details during iteration
msize	number of markers used in compuation of sparseld
mask	is a vector or matrix of TRUE/FALSE specifying if marker should be ignored
GRMlist	is a list providing information about GRM matrix stored in binary files on disk
ve_prior	is a scalar or matrix of prior residual (co)variances
vg_prior	is a scalar or matrix of prior genetic (co)variances

tol	is tolerance, i.e. convergence criteria used in gbayes
nit	is the number of iterations
nburn	is the number of burnin iterations
nthin	is the thinning parameter
nit_local	is the number of local iterations
nit_global	is the number of global iterations
method	specifies the methods used (method="bayesN","bayesA","bayesL","bayesC","bayesR")
algorithm	specifies the algorithm

Value

Returns a list structure including

b	vector or matrix (mxt) of posterior means for marker effects
d	vector or matrix (mxt) of posterior means for marker inclusion probabilities
vb	scalar or vector (t) of posterior means for marker variances
vg	scalar or vector (t) of posterior means for genomic variances
ve	scalar or vector (t) of posterior means for residual variances
rb	matrix (txt) of posterior means for marker correlations
rg	matrix (txt) of posterior means for genomic correlations
re	matrix (txt) of posterior means for residual correlations
pi	vector (1xnmodels) of posterior probabilities for models
h2	vector (1xt) of posterior means for model probability
param	a list current parameters (same information as item listed above) used for restart of the analysis
stat	matrix (mxt) of marker information and effects used for genomic risk scoring

Author(s)

Peter Sørensen

Examples

```
# Simulate data and test functions

W <- matrix(rnorm(100000),nrow=1000)
set1 <- sample(1:ncol(W),5)
set2 <- sample(1:ncol(W),5)
sets <- list(set1,set2)
g <- rowSums(W[,c(set1,set2)])
e <- rnorm(nrow(W),mean=0,sd=1)
y <- g + e
```

```
fitM <- gbayes(y=y, W=W, method="bayesN")
fitA <- gbayes(y=y, W=W, method="bayesA")
fitL <- gbayes(y=y, W=W, method="bayesL")
fitC <- gbayes(y=y, W=W, method="bayesC")
```

`getG` *Get elements from genotype matrix stored in PLINK bedfiles*

Description

Extracts specific rows (based on ids or row numbers) and columns (based on rsids or column numbers) from a genotype matrix stored on disk. The extraction is based on provided arguments such as chromosome number, ids, rsids, etc. Genotypes can be optionally scaled and imputed.

Usage

```
getG(
  Glist = NULL,
  chr = NULL,
  bedfiles = NULL,
  bimfiles = NULL,
  famfiles = NULL,
  ids = NULL,
  rsids = NULL,
  rws = NULL,
  cls = NULL,
  impute = TRUE,
  scale = FALSE
)
```

Arguments

`Glist`	A list structure containing information about genotypes stored on disk.
`chr`	An integer representing the chromosome for which the genotype matrix is to be extracted. It is required.
`bedfiles`	A vector of filenames for the PLINK bed-file.
`bimfiles`	A vector of filenames for the PLINK bim-file.
`famfiles`	A vector of filenames for the PLINK fam-file.
`ids`	A vector of individual IDs for whom the genotype data needs to be extracted.
`rsids`	A vector of SNP identifiers for which the genotype data needs to be extracted.
`rws`	A vector of row numbers to be extracted from the genotype matrix.
`cls`	A vector of column numbers to be extracted from the genotype matrix.

impute	A logical or integer. If TRUE, missing genotypes are replaced with their expected values (2 times the allele frequency). If set to an integer, missing values are replaced by that integer.
scale	A logical. If TRUE, the genotype markers are scaled to have a mean of zero and variance of one.

Details

This function facilitates the extraction of specific genotype data from storage based on various criteria. The extracted genotype data can be optionally scaled or imputed. If rsids are provided that are not found in the 'Glist', a warning is raised.

Value

A matrix with extracted genotypic data. Rows correspond to individuals, and columns correspond to SNPs. Row names are set to individual IDs, and column names are set to rsids.

---

gfilter *Filter genetic marker data based on different quality measures*

Description

Quality control is a critical step for working with summary statistics (in particular for external). Processing and quality control of GWAS summary statistics includes:

- map marker ids (rsids/cpra (chr, pos, ref, alt)) to LD reference panel data - check effect allele (flip EA, EAF, Effect) - check effect allele frequency - thresholds for MAF and HWE - exclude INDELS, CG/AT and MHC region - remove duplicated marker ids - check which build version - check for concordance between marker effect and LD data

External summary statistics format: marker, chr, pos, effect_allele, non_effect_allele, effect_allele_freq, effect, effect_se, stat, p, n

Internal summary statistics format: rsids, chr, pos, a1, a2, af, b, seb, stat, p, n

Usage

```
gfilter(
  Glist = NULL,
  excludeMAF = 0.01,
  excludeMISS = 0.05,
  excludeINFO = NULL,
  excludeCGAT = TRUE,
  excludeINDEL = TRUE,
  excludeDUPS = TRUE,
  excludeHWE = 1e-12,
  excludeMHC = FALSE,
  assembly = "GRCh37"
)
```

Arguments

Glist	A list containing information about the genotype matrix stored on disk.
excludeMAF	A scalar threshold. Exclude markers with a minor allele frequency (MAF) below this threshold. Default is 0.01.
excludeMISS	A scalar threshold. Exclude markers with missingness (MISS) above this threshold. Default is 0.05.
excludeINFO	A scalar threshold. Exclude markers with an info score (INFO) below this threshold. Default is 0.8.
excludeCGAT	A logical value; if TRUE exclude markers if the alleles are ambiguous (i.e., either CG or AT combinations).
excludeINDEL	A logical value; if TRUE exclude markers that are insertions or deletions (INDELs).
excludeDUPS	A logical value; if TRUE exclude markers if their identifiers are duplicated.
excludeHWE	A scalar threshold. Exclude markers where the p-value for the Hardy-Weinberg Equilibrium test is below this threshold. Default is 0.01.
excludeMHC	A logical value; if TRUE exclude markers located within the MHC region.
assembly	A character string indicating the name of the genome assembly (e.g., "GRCh38").

Author(s)

Peter Soerensen

---

glma — *Single marker association analysis using linear models or linear mixed models*

Description

The function glma performs single marker association analysis between genotype markers and the phenotype either based on linear model analysis (LMA) or mixed linear model analysis (MLMA).

The basic MLMA approach involves 1) building a genetic relationship matrix (GRM) that models genome-wide sample structure, 2) estimating the contribution of the GRM to phenotypic variance using a random effects model (with or without additional fixed effects) and 3) computing association statistics that account for this component on phenotypic variance.

MLMA methods are the method of choice when conducting association mapping in the presence of sample structure, including geographic population structure, family relatedness and/or cryptic relatedness. MLMA methods prevent false positive associations and increase power. The general recommendation when using MLMA is to exclude candidate markers from the GRM. This can be efficiently implemented via a leave-one-chromosome-out analysis. Further, it is recommend that analyses of randomly ascertained quantitative traits should include all markers (except for the candidate marker and markers in LD with the candidate marker) in the GRM, except as follows. First, the set of markers included in the GRM can be pruned by LD to reduce running time (with association statistics still computed for all markers). Second, genome-wide significant markers of

large effect should be conditioned out as fixed effects or as an additional random effect (if a large number of associated markers). Third, when population stratification is less of a concern, it may be useful using the top associated markers selected based on the global maximum from out-of sample predictive accuracy.

Usage

```
glma(
  y = NULL,
  X = NULL,
  W = NULL,
  Glist = NULL,
  chr = NULL,
  fit = NULL,
  verbose = FALSE,
  statistic = "mastor",
  ids = NULL,
  rsids = NULL,
  msize = 100,
  scale = TRUE
)
```

Arguments

y	vector or matrix of phenotypes
X	design matrix for factors modeled as fixed effects
W	matrix of centered and scaled genotypes
Glist	list of information about genotype matrix stored on disk
chr	chromosome for which summary statistics are computed
fit	list of information about linear mixed model fit (output from greml)
verbose	is a logical; if TRUE it prints more details during optimization
statistic	single marker test statistic used (currently based on the "mastor" statistics).
ids	vector of individuals used in the analysis
rsids	vector of marker rsids used in the analysis
msize	number of genotype markers used for batch processing
scale	logical if TRUE the genotypes have been scaled to mean zero and variance one

Value

Returns a dataframe (if number of traits = 1) else a list including

coef	single marker coefficients
se	standard error of coefficients
stat	single marker test statistic
p	p-value

Author(s)

Peter Soerensen

References

Chen, W. M., & Abecasis, G. R. (2007). Family-based association tests for genomewide association scans. The American Journal of Human Genetics, 81(5), 913-926.

Loh, P. R., Tucker, G., Bulik-Sullivan, B. K., Vilhjalmsson, B. J., Finucane, H. K., Salem, R. M., ... & Patterson, N. (2015). Efficient Bayesian mixed-model analysis increases association power in large cohorts. Nature genetics, 47(3), 284-290.

Kang, H. M., Sul, J. H., Zaitlen, N. A., Kong, S. Y., Freimer, N. B., Sabatti, C., & Eskin, E. (2010). Variance component model to account for sample structure in genome-wide association studies. Nature genetics, 42(4), 348-354.

Lippert, C., Listgarten, J., Liu, Y., Kadie, C. M., Davidson, R. I., & Heckerman, D. (2011). FaST linear mixed models for genome-wide association studies. Nature methods, 8(10), 833-835.

Listgarten, J., Lippert, C., Kadie, C. M., Davidson, R. I., Eskin, E., & Heckerman, D. (2012). Improved linear mixed models for genome-wide association studies. Nature methods, 9(6), 525-526.

Listgarten, J., Lippert, C., & Heckerman, D. (2013). FaST-LMM-Select for addressing confounding from spatial structure and rare variants. Nature Genetics, 45(5), 470-471.

Lippert, C., Quon, G., Kang, E. Y., Kadie, C. M., Listgarten, J., & Heckerman, D. (2013). The benefits of selecting phenotype-specific variants for applications of mixed models in genomics. Scientific reports, 3.

Zhou, X., & Stephens, M. (2012). Genome-wide efficient mixed-model analysis for association studies. Nature genetics, 44(7), 821-824.

Svishcheva, G. R., Axenovich, T. I., Belonogova, N. M., van Duijn, C. M., & Aulchenko, Y. S. (2012). Rapid variance components-based method for whole-genome association analysis. Nature genetics, 44(10), 1166-1170.

Yang, J., Zaitlen, N. A., Goddard, M. E., Visscher, P. M., & Price, A. L. (2014). Advantages and pitfalls in the application of mixed-model association methods. Nature genetics, 46(2), 100-106.

Bulik-Sullivan, B. K., Loh, P. R., Finucane, H. K., Ripke, S., Yang, J., Patterson, N., ... & Schizophrenia Working Group of the Psychiatric Genomics Consortium. (2015). LD Score regression distinguishes confounding from polygenicity in genome-wide association studies. Nature genetics, 47(3), 291-295.

Examples

```
# Simulate data
W <- matrix(rnorm(1000000), ncol = 1000)
 colnames(W) <- as.character(1:ncol(W))
 rownames(W) <- as.character(1:nrow(W))
y <- rowSums(W[, 1:10]) + rowSums(W[, 501:510]) + rnorm(nrow(W))

# Create model
data <- data.frame(y = y, mu = 1)
fm <- y ~ 0 + mu
```

```
X <- model.matrix(fm, data = data)

# Linear model analyses and single marker association test
stat <- glma(y=y,X=X,W = W)

head(stat)


# Compute GRM
GRM <- grm(W = W)

# Estimate variance components using REML analysis
fit <- greml(y = y, X = X, GRM = list(GRM), verbose = TRUE)

# Single marker association test
stat <- glma(fit = fit, W = W)

head(stat)
```

---

gmap *Finemapping using Bayesian Linear Regression Models*

Description

In the Bayesian multiple regression model, the posterior density of the model parameters depends on the likelihood of the data given the parameters and a prior probability for the model parameters. The choice of the prior for marker effects can influence the type and extent of shrinkage induced in the model.

Usage

```
gmap(
  Glist = NULL,
  stat = NULL,
  sets = NULL,
  models = NULL,
  rsids = NULL,
  ids = NULL,
  mask = NULL,
  lambda = NULL,
  vb = NULL,
  vg = NULL,
  ve = NULL,
  vy = NULL,
  pi = NULL,
```

```
  gamma = NULL,
  mc = 5000,
  h2 = 0.5,
  nub = 4,
  nug = 4,
  nue = 4,
  ssb_prior = NULL,
  ssg_prior = NULL,
  sse_prior = NULL,
  vb_prior = NULL,
  vg_prior = NULL,
  ve_prior = NULL,
  updateB = TRUE,
  updateG = TRUE,
  updateE = TRUE,
  updatePi = TRUE,
  formatLD = "dense",
  checkLD = FALSE,
  shrinkLD = FALSE,
  shrinkCor = FALSE,
  pruneLD = FALSE,
  checkConvergence = FALSE,
  critVe = 3,
  critVg = 3,
  critVb = 3,
  critPi = 3,
  critB = 3,
  critB1 = 0.5,
  critB2 = 3,
  verbose = FALSE,
  eigen_threshold = 0.995,
  cs_threshold = 0.9,
  cs_r2 = 0.5,
  nit = 1000,
  nburn = 100,
  nthin = 1,
  output = "summary",
  method = "bayesR",
  algorithm = "mcmc-eigen",
  seed = 10
)
```

Arguments

`Glist`	A list containing information on genotypic data, including SNPs, chromosomes, positions, and optionally, LD matrices.
`stat`	A data frame or list of summary statistics including effect sizes, standard errors, sample sizes, etc.

`sets` Optional list specifying sets of SNPs for mapping.

`models` Optional list of predefined models for Bayesian regression.

`rsids` Vector of SNP identifiers.

`ids` Vector of sample identifiers.

`mask` Logical matrix indicating SNPs to exclude from analysis.

`lambda` Vector of initial values for penalty parameters in the model.

`vb` Initial value for the marker effect variance (default: NULL).

`vg` Initial value for the genetic variance (default: NULL).

`ve` Initial value for the residual variance (default: NULL).

`vy` Initial value for the phenotypic variance (default: NULL).

`pi` Vector of initial values for pi parameters in the model (default of pi=c(0.999,0.001) for bayesC and pi=c(0.994,0.003,0.002,0.001).

`gamma` Vector of initial values for gamma parameters in the model (default of gamma=c(0,1) for bayesC and gamma=c(0,0.01,0.1,1).

`mc` Number of potentiel genome-wide causal markers for the trait analysed - only used for specification of ssb_prior (default: 5000).

`h2` Heritability estimate (default: 0.5).

`nub`, `nug`, `nue` Degrees of freedom parameters for the priors of marker, genetic, and residual variances, respectively.

`ssb_prior`, `ssg_prior`, `sse_prior`
Priors for the marker, genetic, and residual variances.

`vb_prior`, `vg_prior`, `ve_prior`
Additional priors for marker, genetic, and residual variances (default: NULL).

`updateB`, `updateG`, `updateE`, `updatePi`
Logical values specifying whether to update marker effects, genetic variance, residual variance, and inclusion probabilities, respectively.

`formatLD` Format of LD matrix ("dense" by default).

`checkLD` Logical, whether to check the LD matrix for inconsistencies (default: FALSE).

`shrinkLD`, `shrinkCor`
Logical, whether to apply shrinkage to the LD or correlation matrices (default: FALSE).

`pruneLD` Logical, whether to prune LD matrix (default: FALSE).

`checkConvergence`
Logical, whether to check for convergence of the Gibbs sampler (default: FALSE).

`critVe`, `critVg`, `critVb`, `critPi`, `critB`, `critB1`, `critB2`
Convergence criteria for residual, genetic, and marker variances, inclusion probabilities, and marker effects.

`verbose` Logical, whether to print detailed output for debugging (default: FALSE).

`eigen_threshold`
Threshold for eigenvalues in eigen decomposition (default: 0.995).

`cs_threshold`, `cs_r2`
PIP and r2 thresholds credible set construction (default: cs_threshold=0.9, cs_r2=0.5)

nit	Number of iterations in the MCMC sampler (default: 5000).
nburn	Number of burn-in iterations (default: 500).
nthin	Thinning interval for MCMC (default: 5).
output	Level of output, options include "summary", "full".
method	The regression method to use, options include "blup", "bayesN", "bayesA", "bayesL", "bayesC", "bayesR".
algorithm	Algorithm for MCMC sampling, options include "mcmc", "em-mcmc", "mcmc-eigen".
seed	Random seed for reproducibility (default: 10).

Details

This function implements Bayesian linear regression models to provide unified mapping of genetic variants, estimate genetic parameters (e.g. heritability), and predict disease risk. It is designed to handle various genetic architectures and scale efficiently with large datasets.

Value

Returns a list structure including the following components:

Author(s)

Peter Sørensen

---

gprep *Prepare genotype data for all statistical analyses*

Description

All functions in qgg relies on a simple data infrastructure that takes five main input sources; phenotype data (y), covariate data (X), genotype data (G or Glist), a genomic relationship matrix (GRM or GRMlist) and genetic marker sets (sets).

The genotypes are stored in a matrix (n x m (individuals x markers)) in memory (G) or in a binary file on disk (Glist).

It is only for small data sets that the genotype matrix (G) can stored in memory. For large data sets the genotype matrix has to stored in a binary file on disk (Glist). Glist is as a list structure that contains information about the genotypes in the binary file.

The gprep function prepares the Glist, and is required for downstream analyses of large-scale genetic data. Typically, the Glist is prepared once, and saved as an *.Rdata-file.

The gprep function reads genotype information from binary PLINK files, and creates the Glist object that contains general information about the genotypes such as reference alleles, allele frequencies and missing genotypes, and construct a binary file on the disk that contains the genotypes as allele counts of the alternative allele (memory usage = (n x m)/4 bytes).

The gprep function can also be used to prepare sparse ld matrices. The r2 metric used is the pairwise correlation between markers (allele count alternative allele) in a specified region of the genome. The marker genotype is allele count of the alternative allele which is assumed to be centered and scaled.

The Glist structure is used as input parameter for a number of qgg core functions including: 1) construction of genomic relationship matrices (grm), 2) construction of sparse ld matrices, 3) estimating genomic parameters (greml), 4) single marker association analyses (glma), 5) gene set enrichment analyses (gsea), and 6) genomic prediction from genotypes and phenotypes (gsolve) or genotypes and summary statistics (gscore).

Usage

```
gprep(
  Glist = NULL,
  task = "prepare",
  study = NULL,
  fnBED = NULL,
  ldfiles = NULL,
  bedfiles = NULL,
  bimfiles = NULL,
  famfiles = NULL,
  mapfiles = NULL,
  ids = NULL,
  rsids = NULL,
  assembly = NULL,
  overwrite = FALSE,
  msize = 100,
  r2 = NULL,
  kb = NULL,
  cm = NULL,
  ncores = 1
)
```

Arguments

Glist	A list containing information about the genotype matrix stored on disk.
task	A character string specifying the task to perform. Possible tasks are "prepare" (default), "sparseld", "ldscores", and "geneticmap".
study	The name of the study.
fnBED	Path and filename of the .bed binary file used to store genotypes on disk.
ldfiles	Path and filename of the .ld binary files used for storing the sparse LD matrix on disk.
bedfiles	A vector of filenames for the PLINK bed-files.
bimfiles	A vector of filenames for the PLINK bim-files.
famfiles	A vector of filenames for the PLINK fam-files.
mapfiles	A vector of filenames for the mapfiles.
ids	A vector of individual identifiers used in the study.

rsids	A vector of marker rsids used in the study.
assembly	Character string indicating the name of the assembly.
overwrite	A logical value; if TRUE, the binary genotype/LD file will be overwritten.
msize	Number of markers used in the computation of sparseld.
r2	A threshold value (more context might be beneficial, e.g., threshold for what?).
kb	Size of the genomic region in kilobases (kb).
cm	Size of the genomic region in centimorgans (cm).
ncores	Number of processing cores to be used for genotype processing.

Value

Returns a list structure (Glist) with information about the genotypes.

Author(s)

Peter Soerensen

Examples

```
bedfiles <- system.file("extdata", "sample_chr1.bed", package = "qgg")
bimfiles <- system.file("extdata", "sample_chr1.bim", package = "qgg")
famfiles <- system.file("extdata", "sample_chr1.fam", package = "qgg")

Glist <- gprep(study="Example", bedfiles=bedfiles, bimfiles=bimfiles,
             famfiles=famfiles)
```

---

greml *Genomic rescticted maximum likelihood (GREML) analysis*

Description

The greml function is used for the estimation of genomic parameters (co-variance, heritability and correlation) for linear mixed models using restricted maximum likelihood estimation (REML) and genomic prediction using best linear unbiased prediction (BLUP).

The linear mixed model can account for multiple genetic factors (fixed and random genetic marker effects), adjust for complex family relationships or population stratification and adjust for other non-genetic factors including lifestyle characteristics. Different genetic architectures (infinitesimal, few large and many small effects) is accounted for by modeling genetic markers in different sets as fixed or random effects and by specifying individual genetic marker weights. Different genetic models (e.g. additive and non-additive) can be specified by providing additive and non-additive genomic relationship matrices (GRMs) (constructed using grm). The GRMs can be accessed from the R environment or from binary files stored on disk facilitating the analyses of large-scale genetic data.

The output contains estimates of variance components, fixed and random effects, first and second derivatives of log-likelihood and the asymptotic standard deviation of parameter estimates.

Assessment of predictive accuracy (including correlation and R2, and AUC for binary phenotypes) can be obtained by providing greml with a data frame, or a list that contains sample IDs used in the validation (see examples for details).

Genomic parameters can also be estimated with DMU (http://www.dmu.agrsci.dk/DMU/) if interface ="DMU". This option requires DMU to be installed locally, and the path to the DMU binary files has to be specified (see examples below for details).

Usage

```
greml(
  y = NULL,
  X = NULL,
  GRMlist = NULL,
  GRM = NULL,
  theta = NULL,
  ids = NULL,
  validate = NULL,
  maxit = 100,
  tol = 1e-05,
  bin = NULL,
  ncores = 1,
  wkdir = getwd(),
  verbose = FALSE,
  interface = "R",
  fm = NULL,
  data = NULL
)
```

Arguments

y	is a vector or matrix of phenotypes
X	is a design matrix for factors modeled as fixed effects
GRMlist	is a list providing information about GRM matrix stored in binary files on disk
GRM	is a list of one or more genomic relationship matrices
theta	is a vector of initial values of co-variance for REML estimation
ids	is a vector of individuals used in the analysis
validate	is a data frame or list of individuals used in cross-validation (one column/row for each validation set)
maxit	is the maximum number of iterations used in REML analysis
tol	is tolerance, i.e. convergence criteria used in REML
bin	is the directory for fortran binaries (e.g. DMU binaries dmu1 and dmuai)
ncores	is the number of cores used for the analysis
wkdir	is the working directory used for REML

verbose	is a logical; if TRUE it prints more details during optimization
interface	is used for specifying whether to use R or Fortran implementations of REML
fm	is a formula with model statement for the linear mixed model
data	is a data frame containing the phenotypic observations and fixed factors specified in the model statements

Value

returns a list structure including:

llik	log-likelihood at convergence
theta	covariance estimates from REML
asd	asymptotic standard deviation
b	vector of fixed effect estimates
varb	vector of variances of fixed effect estimates
g	vector or matrix of random effect estimates
e	vector or matrix of residual effects
accuracy	matrix of prediction accuracies (only returned if [validate?] is provided)

Author(s)

Peter Soerensen

References

Lee, S. H., & van der Werf, J. H. (2006). An efficient variance component approach implementing an average information REML suitable for combined LD and linkage mapping with a general complex pedigree. Genetics Selection Evolution, 38(1), 25.

Examples

```
# Simulate data
W <- matrix(rnorm(1000000), ncol = 1000)
 colnames(W) <- as.character(1:ncol(W))
 rownames(W) <- as.character(1:nrow(W))
y <- rowSums(W[, 1:10]) + rowSums(W[, 501:510]) + rnorm(nrow(W))

# Create model
data <- data.frame(y = y, mu = 1)
fm <- y ~ 0 + mu
X <- model.matrix(fm, data = data)

# Compute GRM
GRM <- grm(W = W)

# REML analyses
```

```
fitG <- greml(y = y, X = X, GRM = list(GRM))


# REML analyses and cross validation

# Create marker sets
setsGB <- list(A = colnames(W)) # gblup model
setsGF <- list(C1 = colnames(W)[1:500], C2 = colnames(W)[501:1000]) # gfblup model
setsGT <- list(C1 = colnames(W)[1:10], C2 = colnames(W)[501:510]) # true model

GB <- lapply(setsGB, function(x) {grm(W = W[, x])})
GF <- lapply(setsGF, function(x) {grm(W = W[, x])})
GT <- lapply(setsGT, function(x) {grm(W = W[, x])})

n <- length(y)
fold <- 10
nvalid <- 5

validate <- replicate(nvalid, sample(1:n, as.integer(n / fold)))
cvGB <- greml(y = y, X = X, GRM = GB, validate = validate)
cvGF <- greml(y = y, X = X, GRM = GF, validate = validate)
cvGT <- greml(y = y, X = X, GRM = GT, validate = validate)

cvGB$accuracy
cvGF$accuracy
cvGT$accuracy
```

---

grm *Computing the genomic relationship matrix (GRM)*

Description

The grm function is used to compute a genomic relationship matrix (GRM) based on all, or a subset of marker genotypes. GRM for additive, and non-additive (dominance and epistasis) genetic models can be constructed. The output of the grm function can either be a within-memory GRM object (n x n matrix), or a GRM-list which is a list structure that contains information about the GRM stored in a binary file on the disk.

Usage

```
grm(
  Glist = NULL,
  GRMlist = NULL,
  ids = NULL,
  rsids = NULL,
  rws = NULL,
```

```
  cls = NULL,
  W = NULL,
  method = "add",
  scale = TRUE,
  msize = 100,
  ncores = 1,
  fnG = NULL,
  overwrite = FALSE,
  returnGRM = FALSE,
  miss = NA,
  impute = TRUE,
  pedigree = NULL,
  task = "grm"
)
```

Arguments

Glist	list providing information about genotypes stored on disk
GRMlist	list providing information about GRM matrix stored in binary files on disk
ids	vector of individuals used for computing GRM
rsids	vector marker rsids used for computing GRM
rws	rows in genotype matrix used for computing GRM
cls	columns in genotype matrix used for computing GRM
W	matrix of centered and scaled genotypes
method	indicator of method used for computing GRM: additive (add, default), dominance (dom) or epistasis (epi-pairs or epi-hadamard (all genotype markers))
scale	logical if TRUE the genotypes in Glist has been scaled to mean zero and variance one
msize	number of genotype markers used for batch processing
ncores	number of cores used to compute the GRM
fnG	name of the binary file used for storing the GRM on disk
overwrite	logical if TRUE the binary file fnG will be overwritten
returnGRM	logical if TRUE function returns the GRM matrix to the R environment
miss	the missing code (miss=NA is default) used for missing values in the genotype data
impute	if missing values in the genotype matrix W then mean impute
pedigree	is a dataframe with pedigree information
task	either computation of GRM (task="grm" which is default) or eigenvalue decomposition of GRM (task="eigen")

Value

Returns a genomic relationship matrix (GRM) if returnGRM=TRUE else a list structure (GRMlist) with information about the GRM stored on disk

Author(s)

Peter Soerensen

Examples

```
# Simulate data
W <- matrix(rnorm(1000000), ncol = 1000)
 colnames(W) <- as.character(1:ncol(W))
 rownames(W) <- as.character(1:nrow(W))

# Compute GRM
GRM <- grm(W = W)



# Eigen value decompostion GRM
eig <- grm(GRM=GRM, task="eigen")
```

---

gscore *Genomic scoring based on single marker summary statistics*

Description

Computes genomic predictions using single marker summary statistics and observed genotypes.

Usage

```
gscore(
  Glist = NULL,
  chr = NULL,
  bedfiles = NULL,
  bimfiles = NULL,
  famfiles = NULL,
  stat = NULL,
  fit = NULL,
  ids = NULL,
  scaleMarker = TRUE,
  scaleGRS = TRUE,
  impute = TRUE,
  msize = 100,
  ncores = 1,
  verbose = FALSE
)
```

Arguments

Glist	List of information about genotype matrix. Default is NULL.
chr	Chromosome for which genomic scores is computed. Default is NULL.
bedfiles	Names of the PLINK bed-files. Default is NULL.
bimfiles	Names of the PLINK bim-files. Default is NULL.
famfiles	Names of the PLINK fam-files. Default is NULL.
stat	Matrix of single marker effects. Default is NULL.
fit	Fit object output from gbayes. Default is NULL.
ids	Vector of individuals used in the analysis. Default is NULL.
scaleMarker	Logical; if TRUE the genotype markers are scaled to mean zero and variance one. Default is TRUE.
scaleGRS	Logical; if TRUE the GRS are scaled to mean zero and variance one. Default is TRUE.
impute	Logical; if TRUE, missing genotypes are set to its expected value (2*af where af is allele frequency). Default is TRUE.
msize	Number of genotype markers used for batch processing. Default is 100.
ncores	Number of cores used in the analysis. Default is 1.
verbose	Logical; if TRUE, more details are printed during optimization. Default is FALSE.

Value

Returns the genomic scores based on the provided parameters.

Author(s)

Peter Soerensen

Examples

```
## Plink bed/bim/fam files
bedfiles <- system.file("extdata", paste0("sample_chr",1:2,".bed"), package = "qgg")
bimfiles <- system.file("extdata", paste0("sample_chr",1:2,".bim"), package = "qgg")
famfiles <- system.file("extdata", paste0("sample_chr",1:2,".fam"), package = "qgg")

# Summarize bed/bim/fam files
Glist <- gprep(study="Example", bedfiles=bedfiles, bimfiles=bimfiles, famfiles=famfiles)

# Simulate phenotype
sim <- gsim(Glist=Glist, chr=1, nt=1)

# Compute single marker summary statistics
stat <- glma(y=sim$y, Glist=Glist, scale=FALSE)

# Compute genomic scores
gsc <- gscore(Glist = Glist, stat = stat)
```

gsea *Gene set enrichment analysis*

Description

The function gsea can perform several different gene set enrichment analyses. The general procedure is to obtain single marker statistics (e.g. summary statistics), from which it is possible to compute and evaluate a test statistic for a set of genetic markers that measures a joint degree of association between the marker set and the phenotype. The marker set is defined by a genomic feature such as genes, biological pathways, gene interactions, gene expression profiles etc.

Currently, four types of gene set enrichment analyses can be conducted with gsea; sum-based, count-based, score-based, and our own developed method, the covariance association test (CVAT). For details and comparisons of test statistics consult doi:10.1534/genetics.116.189498.

The sum test is based on the sum of all marker summary statistics located within the feature set. The single marker summary statistics can be obtained from linear model analyses (from PLINK or using the qgg glma approximation), or from single or multiple component REML analyses (GBLUP or GFBLUP) from the greml function. The sum test is powerful if the genomic feature harbors many genetic markers that have small to moderate effects.

The count-based method is based on counting the number of markers within a genomic feature that show association (or have single marker p-value below a certain threshold) with the phenotype. Under the null hypothesis (that the associated markers are picked at random from the total number of markers, thus, no enrichment of markers in any genomic feature) it is assumed that the observed count statistic is a realization from a hypergeometric distribution.

The score-based approach is based on the product between the scaled genotypes in a genomic feature and the residuals from the liner mixed model (obtained from greml).

The covariance association test (CVAT) is derived from the fit object from greml (GBLUP or GFBLUP), and measures the covariance between the total genomic effects for all markers and the genomic effects of the markers within the genomic feature.

The distribution of the test statistics obtained from the sum-based, score-based and CVAT is unknown, therefore a circular permutation approach is used to obtain an empirical distribution of test statistics.

Usage

```
gsea(
  stat = NULL,
  sets = NULL,
  Glist = NULL,
  W = NULL,
  fit = NULL,
  g = NULL,
  e = NULL,
  threshold = 0.05,
  method = "sum",
  nperm = 1000,
```

```
  ncores = 1
)
```

Arguments

stat	vector or matrix of single marker statistics (e.g. coefficients, t-statistics, p-values)
sets	list of marker sets - names corresponds to row names in stat
Glist	list providing information about genotypes stored on disk
W	matrix of centered and scaled genotypes (used if method = cvat or score)
fit	list object obtained from a linear mixed model fit using the greml function
g	vector (or matrix) of genetic effects obtained from a linear mixed model fit (GBLUP of GFBLUP)
e	vector (or matrix) of residual effects obtained from a linear mixed model fit (GBLUP of GFBLUP)
threshold	used if method='hyperg' (threshold=0.05 is default)
method	including sum, cvat, hyperg, score
nperm	number of permutations used for obtaining an empirical p-value
ncores	number of cores used in the analysis

Value

Returns a dataframe or a list including

stat	marker set test statistics
m	number of markers in the set
p	enrichment p-value for marker set

Author(s)

Peter Soerensen

Examples

```
# Simulate data
W <- matrix(rnorm(1000000), ncol = 1000)
colnames(W) <- as.character(1:ncol(W))
rownames(W) <- as.character(1:nrow(W))
y <- rowSums(W[, 1:10]) + rowSums(W[, 501:510]) + rnorm(nrow(W))

# Create model
data <- data.frame(y = y, mu = 1)
fm <- y ~ 0 + mu
X <- model.matrix(fm, data = data)

# Single marker association analyses
```

```
stat <- glma(y=y,X=X,W=W)

# Create marker sets
f <- factor(rep(1:100,each=10), levels=1:100)
sets <- split(as.character(1:1000),f=f)

# Set test based on sums
b2 <- stat[,"stat"]**2
names(b2) <- rownames(stat)
mma <- gsea(stat = b2, sets = sets, method = "sum", nperm = 100)
head(mma)

# Set test based on hyperG
p <- stat[,"p"]
names(p) <- rownames(stat)
mma <- gsea(stat = p, sets = sets, method = "hyperg", threshold = 0.05)
head(mma)


G <- grm(W=W)
fit <- greml(y=y, X=X, GRM=list(G=G), theta=c(10,1))

# Set test based on cvat
mma <- gsea(W=W,fit = fit, sets = sets, nperm = 1000, method="cvat")
head(mma)

# Set test based on score
mma <- gsea(W=W,fit = fit, sets = sets, nperm = 1000, method="score")
head(mma)
```

---

gsim *Genomic simulation*

Description

Simulate Genotype and Phenotype Data

Usage

```
gsim(Glist = NULL, chr = 1, nt = 1, W = NULL, n = 1000, m = 1000, rsids = NULL)
```

Arguments

Glist	A list of information about the genotype matrix. Default is 'NULL'.
chr	The chromosome(s) being used in the simulation. Default is 1.
nt	Number of traits. Default is 1.

W	Matrix of centered and scaled genotypes. Default is 'NULL'.
n	Number of individuals. Default is 1000.
m	Number of markers. Default is 1000.
rsids	A character vector of rsids. Default is 'NULL'.

Details

This function simulates genotype and phenotype data based on the 'Glist', which is information about the genotype matrix.

Value

A list containing:

- y: Phenotypes.
- W: Matrix of centered and scaled genotypes.
- e: Errors.
- g: Genotype effect.
- b0, b1: Coefficients.
- set0, set1: Selected markers.
- causal: Causal markers.

Author(s)

Peter Soerensen

Examples

```
## Plink bed/bim/fam files
bedfiles <- system.file("extdata", paste0("sample_chr",1:2,".bed"), package = "qgg")
bimfiles <- system.file("extdata", paste0("sample_chr",1:2,".bim"), package = "qgg")
famfiles <- system.file("extdata", paste0("sample_chr",1:2,".fam"), package = "qgg")

# Summarize bed/bim/fam files
Glist <- gprep(study="Example", bedfiles=bedfiles, bimfiles=bimfiles, famfiles=famfiles)

# Simulate phenotype
sim <- gsim(Glist=Glist, chr=1, nt=1)
head(sim$y)
head(sim$e)
head(sim$causal)
```

gsolve *Solve linear mixed model equations*

Description

The gsolve function is used for solving of linear mixed model equations. The algorithm used to solve the equation system is based on a Gauss-Seidel (GS) method (matrix-free with residual updates) that handles large data sets.

The linear mixed model fitted can account for multiple traits, multiple genetic factors (fixed or random genetic marker effects), adjust for complex family relationships or population stratification, and adjust for other non-genetic factors including lifestyle characteristics. Different genetic architectures (infinitesimal, few large and many small effects) is accounted for by modeling genetic markers in different sets as fixed or random effects and by specifying individual genetic marker weights.

Usage

```
gsolve(
  y = NULL,
  X = NULL,
  GRM = NULL,
  va = NULL,
  ve = NULL,
  Glist = NULL,
  W = NULL,
  ids = NULL,
  rsids = NULL,
  sets = NULL,
  scale = TRUE,
  lambda = NULL,
  weights = FALSE,
  maxit = 500,
  tol = 1e-05,
  method = "gsru",
  ncores = 1
)
```

Arguments

y	vector or matrix of phenotypes
X	design matrix of fixed effects
GRM	genetic relationship matrix
va	genetic variance
ve	residual variance
Glist	list of information about genotype matrix stored on disk

W	matrix of centered and scaled genotypes
ids	vector of individuals used in the analysis
rsids	vector of marker rsids used in the analysis
sets	list containing marker sets rsids
scale	logical if TRUE the genotypes in Glist will be scaled to mean zero and variance one
lambda	overall shrinkage factor
weights	vector of single marker weights used in BLUP
maxit	maximum number of iterations used in the Gauss-Seidel procedure
tol	tolerance, i.e. the maximum allowed difference between two consecutive iterations of the solver to declare convergence
method	used in solver (currently only methods="gsru": gauss-seidel with resiudal update)
ncores	number of cores used in the analysis

Author(s)

Peter Soerensen

Examples

```
# Simulate data
W <- matrix(rnorm(1000000), ncol = 1000)
 colnames(W) <- as.character(1:ncol(W))
 rownames(W) <- as.character(1:nrow(W))
m <- ncol(W)
causal <- sample(1:ncol(W),50)
y <- rowSums(W[,causal]) + rnorm(nrow(W),sd=sqrt(50))

X <- model.matrix(y~1)

Sg <- 50
Se <- 50
h2 <- Sg/(Sg+Se)
lambda <- Se/(Sg/m)
lambda <- m*(1-h2)/h2

# BLUP of single marker effects and total genomic effects based on Gauss-Seidel procedure
fit <- gsolve( y=y, X=X, W=W, lambda=lambda)
```

ldsc *LD score regression*

Description

The ldsc function is used for LDSC analysis

Usage

```
ldsc(
  Glist = NULL,
  ldscores = NULL,
  sets = NULL,
  method = "regression",
  residual = FALSE,
  z = NULL,
  b = NULL,
  seb = NULL,
  af = NULL,
  stat = NULL,
  tol = 1e-08,
  n = NULL,
  intercept = TRUE,
  what = "h2",
  maxZ2 = NULL,
  SE.h2 = FALSE,
  SE.rg = FALSE,
  blk = 200
)
```

Arguments

Glist	list of information about genotype matrix stored on disk
ldscores	vector of LD scores (optional as LD scores are stored within Glist)
sets	Optional list specifying sets of SNPs for mapping.
method	the regression method to use, options include "regression", "bayesC", "bayesR".
residual	logical if TRUE then add a residual that capture the h2 not explained by the sets
z	matrix of z statistics for n traits
b	matrix of marker effects for n traits if z matrix not is given
seb	matrix of standard errors of marker effects for n traits if z matrix not is given
af	vector of allele frequencies
stat	dataframe with marker summary statistics
tol	smallest value for h2

n	vector of sample sizes for the traits (element i corresponds to column vector i in z matrix)
intercept	logical if TRUE the LD score regression includes intercept
what	either computation of heritability (what="h2") or genetic correlation between traits (what="rg")
maxZ2	maximum value for squared value of z-statistics
SE.h2	logical if TRUE standard errors and significance for the heritability estimates are computed using a block jackknife approach
SE.rg	logical if TRUE standard errors and significance for the genetic correlations are computed using a block jackknife approach
blk	numeric size of the blocks used in the jackknife estimation of standard error (default = 200)

Value

Returns a matrix of heritability estimates when what="h2", and if SE.h2=TRUE standard errors (SE) and significance levels (P) are returned. If what="rg" an n-by-n matrix of correlations is returned where the diagonal elements being h2 estimates. If SE.rg=TRUE a list is returned with n-by-n matrices of genetic correlations, estimated standard errors and significance levels.

Author(s)

Peter Soerensen

Palle Duun Rohde

Examples

```
# Plink bed/bim/fam files
 #bedfiles <- system.file("extdata", paste0("sample_chr",1:2,".bed"), package = "qgg")
 #bimfiles <- system.file("extdata", paste0("sample_chr",1:2,".bim"), package = "qgg")
 #famfiles <- system.file("extdata", paste0("sample_chr",1:2,".fam"), package = "qgg")
 #
 ## Summarize bed/bim/fam files
#Glist <- gprep(study="Example", bedfiles=bedfiles, bimfiles=bimfiles, famfiles=famfiles)

 #
 ## Filter rsids based on MAF, missingness, HWE
 #rsids <-  gfilter(Glist = Glist, excludeMAF=0.05, excludeMISS=0.05, excludeHWE=1e-12)
 #
 ## Compute sparse LD (msize=size of LD window)
 ##ldfiles <- system.file("extdata", paste0("sample_chr",1:2,".ld"), package = "qgg")
##Glist <- gprep(Glist, task="sparseld", msize=200, rsids=rsids, ldfiles=ldfiles, overwrite=TRUE)
 #
 #
 ##Simulate data
 #W1 <- getG(Glist, chr=1, scale=TRUE)
 #W2 <- getG(Glist, chr=2, scale=TRUE)
```

```
#W <- cbind(W1,W2)
#causal <- sample(1:ncol(W),5)

#b1 <- rnorm(length(causal))
#b2 <- rnorm(length(causal))
#y1 <- W[, causal]%*%b1 + rnorm(nrow(W))
#y2 <- W[, causal]%*%b2 + rnorm(nrow(W))

#data1 <- data.frame(y = y1, mu = 1)
#data2 <- data.frame(y = y2, mu = 1)
#X1 <- model.matrix(y ~ 0 + mu, data = data1)
#X2 <- model.matrix(y ~ 0 + mu, data = data2)

## Linear model analyses and single marker association test
#maLM1 <- lma(y=y1, X=X1,W = W)
#maLM2 <- lma(y=y2,X=X2,W = W)
#
## Compute heritability and genetic correlations for trait 1 and 2
#z1 <- maLM1[,"stat"]
#z2 <- maLM2[,"stat"]

#z <- cbind(z1=z1,z2=z2)

#h2 <- ldsc(Glist, z=z, n=c(500,500), what="h2")
#rg <- ldsc(Glist, z=z, n=c(500,500), what="rg")
```

`magma` *Bayesian Multi-marker Analysis of Genomic Annotation (Bayesian MAGMA)*

Description

This function analyzes feature sets using MAGMA or Bayesian methods for association testing. It supports joint or marginal testing, as well as Bayesian linear regression using different priors (‘bayesC‘, ‘bayesR‘).

Usage

```
magma(
  stat = NULL,
  sets = NULL,
  method = "magma",
  type = "joint",
  test = "one-sided",
  pi = 0.001,
  nit = 5000,
```

```
  nburn = 1000
)
```

Arguments

stat	A numeric vector or matrix of summary statistics, where rows represent features and columns represent phenotypes.
sets	A list of feature sets (e.g., genes, SNPs) to be analyzed.
method	A string specifying the method to use. Options are ‘"magma"‘, ‘"blr"‘, ‘"bayesC"‘, or ‘"bayesR"‘. Default is ‘"magma"‘.
type	A string specifying the type of analysis to perform. Options are ‘"joint"‘ (default) or ‘"marginal"‘. Only used with ‘method = "magma"‘.
test	A string specifying the statistical test. Options are ‘"one-sided"‘ (default) or ‘"two-sided"‘. Only used with ‘method = "magma"‘.
pi	A numeric value specifying the proportion of non-zero effects. Used for Bayesian methods. Default is ‘0.001‘.
nit	An integer specifying the number of iterations for Bayesian methods. Default is ‘5000‘.
nburn	An integer specifying the number of burn-in iterations for Bayesian methods. Default is ‘1000‘.

Details

The function uses either the MAGMA approach for set-based testing or Bayesian linear regression to estimate effect sizes and probabilities of association for feature sets. For Bayesian methods, a spike-and-slab prior is applied.

The ‘stat‘ input must have row names corresponding to feature identifiers. The ‘sets‘ input must be a named list, where each element corresponds to a feature set.

Value

A data frame or list with analysis results.

---

mtadj *Adjustment of marker effects using correlated trait information*

Description

The ‘mtadj‘ function uses selection index theory to determine the optimal weights across ‘n‘ traits. These weights are then used to adjust marker effects by ‘n‘ correlated traits. More details can be found [here](https://www.nature.com/articles/s41467-017-02769-6).

Usage

```
mtadj(
  h2 = NULL,
  rg = NULL,
  stat = NULL,
  b = NULL,
  z = NULL,
  n = NULL,
  mtotal = NULL,
  meff = 60000,
  method = "ols",
  statistics = "z"
)
```

Arguments

h2	A vector of heritability estimates.
rg	An n-by-n matrix of genetic correlations.
stat	A dataframe containing marker summary statistics.
b	A matrix of marker effects.
z	A matrix of z-scores.
n	A vector indicating the sample size used to estimate marker effects for each trait.
mtotal	Total number of markers.
meff	Effective number of uncorrelated genomic segments (default = 60,000).
method	Method to estimate marker effects. Can be "OLS" (ordinary least square, default) or "BLUP" (best linear unbiased prediction).
statistics	Specifies which kind of statistics ("b" or "z") should be used in the analysis.

Value

A matrix of adjusted marker effects for each trait.

Author(s)

Palle Duun Rohde and Peter Soerensen

Examples

```
#bedfiles <- system.file("extdata", "sample_22.bed", package = "qgg")
#bimfiles <- system.file("extdata", "sample_22.bim", package = "qgg")
#famfiles <- system.file("extdata", "sample_22.fam", package = "qgg")
#Glist <- gprep(study="1000G", bedfiles=bedfiles, bimfiles=bimfiles,famfiles=famfiles)
#Glist <- gprep(Glist, task="sparseld",  msize=200)
#
##Simulate data
#set.seed(23)
#
```

```
#W <- getG(Glist, chr=1, scale=TRUE)
#causal <- sample(1:ncol(W),50)
#set1 <- c(causal, sample(c(1:ncol(W))[-causal],10))
#set2 <- c(causal, sample(c(1:ncol(W))[-set1],10))
#
#b1 <- rnorm(length(set1))
#b2 <- rnorm(length(set2))
#y1 <- W[, set1]%*%b1 + rnorm(nrow(W))
#y2 <- W[, set2]%*%b2 + rnorm(nrow(W))
#
## Create model
#data1 <- data.frame(y = y1, mu = 1)
#data2 <- data.frame(y = y2, mu = 1)
#X1 <- model.matrix(y ~ 0 + mu, data = data1)
#X2 <- model.matrix(y ~ 0 + mu, data = data2)
#
## Linear model analyses and single marker association test
#maLM1 <- glma(y=y1, X=X1,W = W)
#maLM2 <- glma(y=y2,X=X2,W = W)
#
## Compute genetic parameters
#z1 <- maLM1[,"stat"]
#z2 <- maLM2[,"stat"]
#
#z <- cbind(z1=z1,z2=z2)
#
#h2 <- ldsc(Glist, z=z, n=c(500,500), what="h2")
#rg <- ldsc(Glist, z=z, n=c(500,500), what="rg")
#
## Adjust summary statistics using estimated genetic parameters
#b <- cbind(b1=maLM1[,"b"],b2=maLM2[,"b"])
#bm <- mtadj( h2=h2, rg=rg, b=b, n=c(500,500), method="ols")
```

---

pops *Bayesian Polygenic Prioritisation Scoring (Bayesian POPS)*

Description

This function performs Polygenic Prioritisation Scoring (POPS) using Bayesian regression (‘bayesC‘ or ‘bayesR‘) or ridge regression (‘rr‘). It maps features to sets, performs optional feature selection based on p-value thresholds, and calculates predictive scores for prioritisation.

Usage

```
pops(
  stat = NULL,
  sets = NULL,
  validate = NULL,
```

```
  threshold = NULL,
  method = "bayesC",
  pi = 0.001,
  nit = 5000,
  nburn = 1000,
  updateB = TRUE,
  updateE = TRUE,
  updatePi = TRUE,
  updateG = TRUE
)
```

Arguments

stat	A numeric vector or matrix of summary statistics (e.g., phenotypic values or effect sizes), where rows represent features (e.g., SNPs) and columns represent traits. Required.
sets	A list of feature sets (e.g., genes or SNP groups) to map to the rows of 'stat'. Required.
validate	An optional validation set. If provided, cross-validation results are returned instead of fitting the model.
threshold	A numeric value specifying a p-value threshold for feature selection. If provided, only features with p-values below this threshold are included in the model.
method	A string specifying the regression method. Options are '"bayesC"' (default), '"bayesR"', or '"rr"' (ridge regression).
pi	A numeric value specifying the proportion of non-zero effects for Bayesian methods. Default is '0.001'.
nit	An integer specifying the number of iterations for Bayesian methods. Default is '5000'.
nburn	An integer specifying the number of burn-in iterations for Bayesian methods. Default is '1000'.
updateB	A logical value indicating whether to update marker effects in Bayesian methods. Default is 'TRUE'.
updateE	A logical value indicating whether to update residual variances in Bayesian methods. Default is 'TRUE'.
updatePi	A logical value indicating whether to update the proportion of non-zero effects in Bayesian methods. Default is 'TRUE'.
updateG	A logical value indicating whether to update the genomic variances in Bayesian methods. Default is 'TRUE'.

Value

A matrix of predicted prioritisation scores ('ypred') for each feature, ordered by their predictive values. If a validation set is provided, cross-validation results are returned instead.

vegas *Perform VEGAS Gene-Based Association Analysis*

Description

This function performs VEGAS (Versatile Gene-based Association Study) to analyze gene-level associations using marker statistics and linkage disequilibrium (LD) structure from a reference panel.

Usage

```
vegas(
  Glist = NULL,
  sets = NULL,
  stat = NULL,
  p = NULL,
  threshold = 1e-10,
  tol = 1e-07,
  minsize = 2,
  verbose = FALSE
)
```

Arguments

Glist	A list containing genomic information, such as LD matrices or genotype data. Required.
sets	A list of sets (e.g., genes with their associated markers) to analyze. Required.
stat	A data frame containing marker-level statistics. Must include ‘rsids‘ (marker IDs) and ‘p‘ (p-values).
p	A numeric matrix of p-values for markers across multiple studies. If provided, ‘stat‘ should be NULL.
threshold	A numeric value specifying the lower bound for p-values to avoid numerical issues. Default is ‘1e-10‘.
tol	A numeric value specifying the tolerance for eigenvalues in LD matrices. Default is ‘1e-7‘.
minsize	An integer specifying the minimum number of markers required for a set to be analyzed. Default is ‘2‘.
verbose	A logical value indicating whether to print progress messages. Default is ‘FALSE‘.

Details

The function uses marker-level statistics to compute gene-level association statistics, accounting for LD structure among markers. The LD structure is retrieved from ‘Glist‘, which should include precomputed LD matrices or genotype data for the markers.

Two modes are supported: - **‘stat‘ Mode**: Uses marker statistics (e.g., p-values) from a single study to compute gene-level statistics. - **‘p‘ Mode**: Uses marker p-values across multiple studies for meta-analysis of gene-level statistics.

Value

A data frame with the results

Index

acc, 2
adjStat, 3

gbayes, 4
getG, 8
gfilter, 9
glma, 10
gmap, 13
gprep, 16
greml, 18
grm, 21
gscore, 23
gsea, 25
gsim, 27
gsolve, 29

ldsc, 31

magma, 33
mtadj, 34

pops, 36

vegas, 38