Package: statgenGWAS (via r-universe)

October 22, 2026

Type Package

Title Genome Wide Association Studies

Version 1.0.14

Date 2026-09-22

Description Fast single trait Genome Wide Association Studies (GWAS) following the method described in Kang et al. (2010), <doi:10.1038/ng.548>. One of a series of statistical genetic packages for streamlining the analysis of typical plant breeding experiments developed by Biometris.

License GPL-3

Encoding UTF-8

LazyData true

Depends R (>= 3.6)

Imports data.table, ggplot2 (>= 3.0.0), LMMsolver, methods, rlang, Rcpp, sommer (>= 4.4.1)

Suggests knitr, rmarkdown, officer, tinytest, snpStats, vcfR

VignetteBuilder knitr

LinkingTo Rcpp, RcppArmadillo

URL https://biometris.github.io/statgenGWAS/index.html, https://github.com/Biometris/statgenGWAS/

BugReports https://github.com/Biometris/statgenGWAS/issues

Config/roxygen2/version 8.1.0

NeedsCompilation yes

Author Bart-Jan van Rossum [aut, cre] (ORCID: <https://orcid.org/0000-0002-8673-2514>), Willem Kruijer [aut] (ORCID: <https://orcid.org/0000-0001-7179-1733>), Fred van Eeuwijk [ctb] (ORCID: <https://orcid.org/0000-0003-3672-2921>), Martin Boer [ctb] (ORCID: <https://orcid.org/0000-0002-1879-4588>), Marcos Malosetti [ctb] (ORCID: <https://orcid.org/0000-0002-8150-1397>), Daniela

Bustos-Korts [ctb] (ORCID:
<https://orcid.org/0000-0003-3827-6726>), Emilie Millet [ctb]
(ORCID: <https://orcid.org/0000-0002-2913-4892>), Joao Paulo
[ctb] (ORCID: <https://orcid.org/0000-0002-4180-0763>), Maikel
Verouden [ctb] (ORCID:
<https://orcid.org/0000-0002-4893-3323>), Ron Wehrens [ctb]
(ORCID: <https://orcid.org/0000-0002-8798-5599>), Choazhi Zheng
[ctb] (ORCID: <https://orcid.org/0000-0001-6030-3933>)

Maintainer Bart-Jan van Rossum <bart-jan.vanrossum@wur.nl>

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

Date/Publication 2026-09-22 09:50:02 UTC

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

RemoteRef HEAD

RemoteSha 8c61688899668b76ca13ae25c05e5566776f6cbd

Contents

codeMarkers 2
dropsData 5
gData 7
kinship 9
plot.gData 10
plot.GWAS 12
readPLINK 15
readVcf 16
runSingleTraitGwas 17
summary.gData 21
summary.GWAS 22

Index **23**

`codeMarkers` *Code and impute markers*

Description

`codeMarkers` codes markers in a `gData` object and optionally performs imputation of missing values as well.
The function performs the following steps:

1. replace strings in `naStrings` by `NA`.
2. remove genotypes with a fraction of missing values higher than `nMissGeno`.
3. remove SNPs with a fraction of missing values higher than `nMiss`.

4. recode SNPs to numerical values.
5. remove SNPs with a minor allele frequency lower than MAF.
6. optionally remove duplicate SNPs.
7. optionally impute missing values.
8. repeat steps 5. and 6. if missing values are imputed.

Usage

```
codeMarkers(
  gData,
  refAll = "minor",
  nMissGeno = 1,
  nMiss = 1,
  MAF = NULL,
  MAC = NULL,
  removeDuplicates = TRUE,
  keep = NULL,
  impute = TRUE,
  imputeType = c("random", "fixed", "beagle"),
  fixedValue = NULL,
  naStrings = NA,
  verbose = FALSE
)
```

Arguments

gData	An object of class gData containing at least markers.
refAll	A character string indicating the reference allele used when recoding markers. If "minor", then the recoding is done using the minor allele as reference allele. Alternatively a single character can be supplied as a reference allele for the whole set of SNPs, or a character vector with a reference allele per SNP.
nMissGeno	A numerical value between 0 and 1. Genotypes with a fraction of missing values higher than nMissGeno will be removed. Genotypes with only missing values will always be removed.
nMiss	A numerical value between 0 and 1. SNPs with a fraction of missing values higher than nMiss will be removed. SNPs with only missing values will always be removed.
MAF	A numerical value between 0 and 1. SNPs with a Minor Allele Frequency (MAF) below this value will be removed. Only one of MAF and MAC may be specified.
MAC	A numerical value. SNPs with Minor Allele Count (MAC) below this value will be removed. Only one of MAF and MAC may be specified.
removeDuplicates	Should duplicate SNPs be removed?
keep	A vector of SNPs that should never be removed in the whole process.

impute	Should imputation of missing values be done?
imputeType	A character string indicating what kind of imputation of values should be done. • fixed - missing values will be replaced by a given fixed value. • random - missing values will be replaced by a random value calculated using allele frequencies per SNP. • beagle - missing values will be imputed using beagle software, version 5.2. Beagle only accepts integers as map positions. If you use this option, please cite the original papers in your publication (see references).
fixedValue	A numerical value used for replacing missing values in case inputType is fixed.
naStrings	A character vector of strings to be treated as NA.
verbose	Should a summary of the performed steps be printed?

Value

A copy of the input gData object with markers replaced by coded and imputed markers.

References

S R Browning and B L Browning (2007) Rapid and accurate haplotype phasing and missing data inference for whole genome association studies by use of localized haplotype clustering. Am J Hum Genet 81:1084-1097. doi:10.1086/521987

Examples

```
## Create markers
markers <- matrix(c(
"AA",   "AB",   "AA",   "BB",   "BA",   "AB",   "AA",   "AA",   NA,  "AA",
"AA",   "AA",   "BB",   "BB",   "AA",   "AA",   "BB",   "AA",   NA,  "AA",
"AA",   "BA",   "AB",   "BB",   "AB",   "AB",   "AA",   "BB",   NA,  "AA",
"AA",   "AA",   "BB",   "BB",   "AA",   "AA",   "AA",   "AA",   NA,  "AA",
"AA",   "AA",   "BB",   "BB",   "AA",   "BB",   "BB",   "BB",  "AB", "AA",
"AA",   "AA",   "BB",   "BB",   "AA",    NA,    "BB",   "AA",   NA,  "AA",
"AB",   "AB",   "BB",   "BB",   "BB",   "AA",   "BB",   "BB",   NA,  "AB",
"AA",   "AA",    NA,    "BB",    NA,    "AA",   "AA",   "AA",  "AA", "AA",
"AA",    NA,     NA,    "BB",   "BB",   "BB",   "BB",   "BB",  "AA", "AA",
"AA",    NA,    "AA",   "BB",   "BB",   "BB",   "AA",   "AA",   NA,  "AA"),
ncol = 10, byrow = TRUE, dimnames = list(paste0("IND", 1:10),
paste0("SNP", 1:10)))

## create object of class 'gData'.
gData <- createGData(geno = markers)

## Code markers by minor allele, no imputation.
gDataCoded1 <- codeMarkers(gData = gData, impute = FALSE)

## Code markers by reference alleles, impute missings by fixed value.
gDataCoded2 <- codeMarkers(gData = gData,
                           refAll = rep(x = c("A", "B"), times =  5),
```

```
                              impute = TRUE, imputeType = "fixed",
                              fixedValue = 1)

## Code markers by minor allele, impute by random value.
gDataCoded3 <- codeMarkers(gData = gData, impute = TRUE,
                              imputeType = "random")
```

`dropsData` *DROPS data sets*

Description

This dataset comes from the European Union project DROPS (DROught-tolerant yielding PlantS). A panel of 256 maize hybrids was grown with two water regimes (irrigated or rainfed), in seven fields in 2012 and 2013, respectively, spread along a climatic transect from western to eastern Europe, plus one site in Chile in 2013. This resulted in 28 experiments defined as the combination of one year, one site and one water regime, with two and three repetitions for rainfed and irrigated treatments, respectively. A detailed environmental characterisation was carried out, with hourly records of micrometeorological data and soil water status, and associated with precise measurement of phenology. Grain yield and its components were measured at the end of the experiment.
10 experiments have been selected from the full data set, two for each of the five main environmental scenarios that were identified in the data. The scenarios have been added to the data as well as a classification of the genotypes in four genetic groups.
The main purpose of this dataset consists in using the environmental characterization to quantify the genetic variability of maize grain yield in response to the environmental drivers for genotype-by-environment interaction. For instance, allelic effects at QTLs identified over the field network are consistent within a scenario but largely differ between scenarios.

The data is split in three separate data.frames.

dropsMarkers This data.frame contains the 50K genotyping matrix coded in allelic dose (012) filtered and imputed. Genotyping of 41,722 loci on 246 parental lines were obtained using 50K Illumina Infinium HD arrays (Ganal et al., 2011). Genotype were coded in allelic dose with 0 for the minor allele, 1 for the heterozygote, and 2 for the major allele. Genotype were filtered (MAF > 1%) and missing data imputed using Beagle v3.
A data.frame with 246 rows and 41723 columns.

Ind name of the genotype

SYMN83 to PZE-110111485 coded QTLs

dropsMap This data.frame contains the description of the 41,722 loci genotyped by 50K Illumina Infinium Array on the 246 lines.
A data.frame with 41722 rows and 5 columns.

SNP.names name of the SNP

Chromosome number of the B73 reference genome V2

Position position on the B73 reference genome V2 in basepairs

allele1 first original allele (A, T, G or C)

allele2 second original allele (A, T, G or C)

dropsPheno This data.frame contains the genotypic means (Best Linear Unbiased Estimators, BLUEs), with one value per experiment (Location × year × water regime) per genotype.
A data.frame with 2460 rows and 19 columns.

Experiment experiments ID described by the three first letters of the city's name followed by the year of experiment and the water regime with W for watered and R for rain-fed.

parent1 identifier of donor dent line

Code_ID, Variety_ID, Accession_ID identifier of the genotype

geno.panel project in which the genetic material was generated

grain.yield genotypic mean for yield adjusted at 15% grain moisture, in ton per hectare (t ha^-1)

grain.number genotypic mean for number of grain per square meter

grain.weight genotypic mean for individual grain weight in milligram (mg)

anthesis genotypic mean for male flowering (pollen shed), in thermal time cumulated since emergence (d_20°C)

silking genotypic mean for female flowering (silking emergence), in thermal time cumulated since emergence (d_20°C)

plant.height genotypic mean for plant height, from ground level to the base of the flag leaf (highest) leaf in centimeter (cm)

tassel.height genotypic mean for plant height including tassel, from ground level to the highest point of the tassel in centimeter (cm)

ear.height genotypic mean for ear insertion height, from ground level to ligule of the highest ear leaf in centimeter (cm)

year year in which the experiment was performed

loc location where the experiment was performed, a three letter abbreviation

scenarioWater water scenario for the experiment, well watered (WW) or water deficit (WD)

scenarioTemp temperature scenario for the experiment, Cool, Hot or Hot(Day)

scenarioFull the full scenario for the experiment, a combination of scenarioWater and scenarioTemp

geneticGroup the genetic group to which the genotype belongs

Note

From the source data, the experiments from 2011 have been removed since these do not contain all genotypes. Also the experiment Gra13W has been removed.

Source

doi:10.15454/IASSTN

References

Millet, E. J., Pommier, C., et al. (2019). A multi-site experiment in a network of European fields for assessing the maize yield response to environmental scenarios - Data set. doi:10.15454/IASSTN

Ganal MW, et al. (2011) A Large Maize (Zea mays L.) SNP Genotyping Array: Development and Germplasm Genotyping, and Genetic Mapping to Compare with the B73 Reference Genome. PLoS ONE 6(12): e28334. doi:10.1371/journal.pone.0028334

`gData` *S3 Class gData*

Description

`createGData` creates an object of S3 class gData with genotypic and phenotypic data for usage in further analysis. All input to the function is optional, however at least one input should be provided. It is possible to provide an existing `gData` object as additional input in which case data is added to this object. Existing data will be overwritten with a warning.

Usage

```
createGData(
  gData = NULL,
  geno = NULL,
  map = NULL,
  kin = NULL,
  pheno = NULL,
  covar = NULL
)
```

Arguments

`gData`	An optional gData object to be modified. If `NULL`, a new gData object is created.
`geno`	A matrix or data.frame with genotypes in the rows and markers in the columns. A matrix from the `matrix` in the base package may be provided as well as as matrix from the Matrix package. If no row names are provided, they are taken from `pheno` (if supplied and dimension matches). If no column names are provided, the row names from `map` are used (if supplied and dimension matches).
`map`	A data.frame with columns `chr` for chromosome and `pos` for position. Positions can be in base pair (bp) or centimorgan (cM). They should not be cumulative over the chromosomes. Other columns are ignored. Marker names should be in the row names. These should match the marker names in `geno` (if supplied).
`kin`	A kinship matrix or list of kinship matrices with genotype in rows and colums. These matrices can be from the `matrix` class, as defined in the base package, or from the `dsyMatrix` class, the class of symmetric matrices in the Matrix package. The genotypes should be identical to the genotypes in `geno`. If a list of kinship matrices is provided these are supposed to be chromosome specific matrices. In that case their names should match the names of the chromosomes in `map`. If no names are provided, the number of matrices should match the number of chromosomes in `map` in which case default names are provided.

pheno	A data.frame or a list of data.frames with phenotypic data, with genotypes in the first column genotype and traits in the following columns. The trait columns should be numerical columns only. A list of data.frames can be used for replications, i.e. different trials.
covar	A data.frame with extra covariates per genotype. Genotypes should be in the rows.

Value

An object of class gData with the following components:

map	a data.frame containing map data. Map is sorted by chromosome and position.
markers	a matrix containing marker information.
pheno	a list of data.frames containing phenotypic data.
kinship	a kinship matrix.
covar	a data.frame with extra covariates.

Author(s)

Bart-Jan van Rossum

See Also

summary.gData

Examples

```
set.seed(1234)
## Create genotypic data.
geno <- matrix(sample(x = c(0, 1, 2), size = 15, replace = TRUE), nrow = 3)
dimnames(geno) <- list(paste0("G", 1:3), paste0("M", 1:5))

## Construct map.
map <- data.frame(chr = c(1, 1, 2, 2, 2), pos = 1:5,
                  row.names = paste0("M", 1:5))

## Compute kinship matrix.
kin <- kinship(X = geno, method = "IBS")

## Create phenotypic data.
pheno <- data.frame(paste0("G", 1:3),
                    matrix(rnorm(n = 12, mean = 50, sd = 5), nrow = 3),
                    stringsAsFactors = FALSE)
dimnames(pheno) = list(paste0("G", 1:3), c("genotype", paste0("T", 1:4)))

## Combine all data in gData object.
gData <- createGData(geno = geno, map = map, kin = kin, pheno = pheno)
summary(gData)

## Construct covariate.
```

```
covar <- data.frame(C1 = c("a", "a", "b"), row.names = paste0("G", 1:3))

## Compute alternative kinship matrix.
kin2 <- kinship(X = geno, method = "astle")

## Add covariates to previously created gData object and overwrite
## current kinship matrix by newly computed one.
gData2 <- createGData(gData = gData, kin = kin2, covar = covar)
```

`kinship` *Functions for calculating kinship matrices*

Description

A collection of functions for calculating kinship matrices using different algorithms. The following algorithms are included: astle (Astle and Balding, 2009), Identity By State (IBS) and VanRaden (VanRaden, 2008) for marker matrices. For method identity an identity kinship matrix is returned.

Usage

```
kinship(
  X,
  method = c("astle", "IBS", "vanRaden", "identity"),
  MAF = NULL,
  denominator = NULL
)
```

Arguments

`X`	An n x m marker matrix with genotypes in the rows (n) and markers in the columns (m).
`method`	The method used for computing the kinship matrix.
`MAF`	The minor allele frequency (MAF) threshold used in kinship computation. A numerical value between 0 and 1. SNPs with MAF below this value are not taken into account when computing the kinship. If NULL all markers are used regardless of their allele frequency.
`denominator`	A numerical value. See details.

Value

An n x n kinship matrix.

Marker matrices

In all algorithms the input matrix `X` is first cleaned, i.e. markers with a variance of 0 are excluded from the calculation of the kinship matrix. Then some form of scaling is done which differs per algorithm. This gives a scaled matrix `Z`. The matrix $ZZ^t/denominator$ is returned. By default the denominator is equal to the number of columns in `Z` for `astle` and `IBS` and $2 * p * (1 - p)$ where $p = colSums(X)/(2 * nrow(X))$ for `vanRaden`. This denominator can be overwritten by the user, e.g. when computing kinship matrices by splitting `X` in smaller matrices and then adding the results together in the end.

References

Astle, William, and David J. Balding. 2009. "Population Structure and Cryptic Relatedness in Genetic Association Studies." Statistical Science 24 (4): 451–71. doi:10.1214/09sts307.

VanRaden P.M. (2008) Efficient methods to compute genomic predictions. Journal of Dairy Science 91 (11): 4414–23. doi:10.3168/jds.20070980.

Examples

```
## Create example matrix.
M <- matrix(c(1, 1, 1, 0, 0, 0, 1, 1, 0, 0, 0, 0, 1, 0, 1, 1), nrow = 4)

## Compute kinship matrices using different methods.
kinship(M, method = "astle")
kinship(M, method = "IBS")
kinship(M, method = "vanRaden")

## Only use markers with a Minor Allele Frequency of 0.3 or more.
kinship(M, method = "astle", MAF = 0.3)

## Compute kinship matrix using astle and balding method with denominator 2.
kinship(M, method = "astle", denominator = 2)
```

---

plot.gData *Plot function for the class* gData

Description

Creates a plot of the genetic map in an object of S3 class `gData`. A plot of the genetic map showing the length of the chromosomes and the positions of the markers in the genetic map is created.

Usage

```
## S3 method for class 'gData'
plot(x, ..., highlight = NULL, title = NULL, output = TRUE)
```

Arguments

x	An object of class gData.
...	Not used.
highlight	A data.frame with at least columns chr and pos, containing marker positions that should be highlighted in the plot. If a column "name" is present that is used for annotation, otherwise the highlighted markers are annotated as chr\@pos#'
title	A character string, the title of the plot.
output	Should the plot be output to the current device? If FALSE, only a ggplot object is invisibly returned.

Value

An object of class ggplot is invisibly returned.

Examples

```
set.seed(1234)
## Create genotypic data.
geno <- matrix(sample(x = c(0, 1, 2), size = 15, replace = TRUE), nrow = 3)
dimnames(geno) <- list(paste0("G", 1:3), paste0("M", 1:5))

## Construct map.
map <- data.frame(chr = c(1, 1, 2, 2, 2), pos = 1:5,
                  row.names = paste0("M", 1:5))

## Compute kinship matrix.
kin <- kinship(X = geno, method = "IBS")

## Create phenotypic data.
pheno <- data.frame(paste0("G", 1:3),
                    matrix(rnorm(n = 12, mean = 50, sd = 5), nrow = 3),
                    stringsAsFactors = FALSE)
dimnames(pheno) = list(paste0("G", 1:3), c("genotype", paste0("T", 1:4)))

## Combine all data in gData object.
gData <- createGData(geno = geno, map = map, kin = kin, pheno = pheno)

## Plot genetic map.
plot(gData)

## Plot genetic map. Highlight first marker in map.
plot(gData, highlight = map[1, ])
```

plot.GWAS *Plot function for the class* GWAS

Description

Creates a plot of an object of S3 class GWAS. The following types of plot can be made:

- a manhattan plot, i.e. a plot of LOD-scores per SNP
- a QQ plot of observed LOD-scores versus expected LOD-scores
- a qtl plot of effect sizes and directions for multiple traits

Manhattan plots and QQ plots are made for a single trait which should be indicated using the parameter trait. If the analysis was done for only one trait, it is detected automatically. The qtl plot will plot all traits analyzed.
See details for a detailed description of the plots and the plot options specific to the different plots.

Usage

```
## S3 method for class 'GWAS'
plot(
  x,
  ...,
  plotType = c("manhattan", "qq", "qtl"),
  trial = NULL,
  trait = NULL,
  title = NULL,
  output = TRUE
)
```

Arguments

x	An object of class GWAS.
...	Further arguments to be passed on to the actual plotting functions.
plotType	A character string indicating the type of plot to be made. One of "manhattan", "qq" and "qtl".
trial	A character string or numeric index indicating for which trial the plot should be made. If x only contains results for one trial, trial may be NULL.
trait	A character string indicating for which trait the results should be plotted. For type "qtl" all traits are plotted. If x only contains results for one trait, trait may be NULL.
title	A character string, the title of the plot.
output	Should the plot be output to the current device? If FALSE, only a list of ggplot objects is invisibly returned.

Manhattan Plot

A LOD-profile of all marker positions and corresponding LOD-scores is plotted. Significant markers are highlighted with red dots. By default these are taken from the result of the GWAS analysis however the LOD-threshold for significant parameters may be modified using the parameter `yThr`. The threshold is plotted as a horizontal line. If there are previously known marker effect, false positives and true negatives can also be marked.
Extra parameter options:

`xLab` A character string, the x-axis label. Default = `"Chromosomes"`

`yLab` A character string, the y-axis label. Default = `-log10(p)`

`effects` A character vector, indicating which SNPs correspond to a real (known) effect. Used for determining true/false positives and false negatives. True positives are colored green, false positives orange and false negatives yellow.

`colPalette` A color palette used for plotting. Default coloring is done by chromosome, using black and grey.

`yThr` A numerical value for the LOD-threshold. The value from the GWAS analysis is used as default.

`signLwd` A numerical value giving the thickness of the points that are false/true positives/negatives. Default = 0.6

`lod` A positive numerical value. For the SNPs with a LOD-value below this value, only 5% is plotted. The chance of a SNP being plotted is proportional to its LOD-score. This option can be useful when plotting a large number of SNPs. The 5% of SNPs plotted is selected randomly. For reproducible results use set.seed before calling the function.

`chr` A vector of chromosomes to be plotted. By default, all chromosomes are plotted. Using this option allows restricting the plot to a subset of chromosomes.

`startPos` A numerical value indicating the start position for the plot. Using this option allows restricting the plot to a part of a selected chromosome. Only used if exactly one chromosome is specified in `chr`.

`endPos` A numerical value indicating the end position for the plot. Using this option allows restricting the plot to a part of a selected chromosome. Only used if exactly one chromosome is specified in `chr`.

QQ-Plot

From the LOD-scores calculated in the GWAS analysis, a QQ-plot is generated with observed LOD-scores versus expected LOD-scores. Code is adapted from Segura et al. (2012).

QTL Plot

A plot of effect sizes for the significant SNPs found in the GWAS analysis is created. Each horizontal line contains QTLs of one trait, phenotypic trait or trial. Optionally, vertical white lines can indicate chromosome subdivision, genes of interest, known QTL, etc. Circle diameters are proportional to the absolute value of allelic effect. Colors indicate the direction of the effect: green when the allele increases the trait value, and blue when it decreases the value.
Extra parameter options:

`normalize` Should the snpEffect be normalized? Default = `FALSE`

`sortData` Should the data be sorted before plotting? Either `FALSE`, if no sorting should be done, or a character string indicating the data column to use for sorting. This should be a numerical column. Default = `FALSE`

`binPositions` An optional data.frame containing at leasts two columns, chr(omosome) and pos(ition). Vertical lines are plotted at those positions. Default = `NULL`

`printVertGrid` Should default vertical grid lines be plotted. Default = `TRUE`

`yLab` A character string, the y-axis label. Default = `"Traits"`

`yThr` A numerical value for the LOD-threshold. The value from the GWAS analysis is used as default.

`chr` A vector of chromosomes to be plotted. By default all chromosomes are plotted. Using this option this can be restricted to a subset of chromosomes.

`exportPptx` Should the plot be exported to a .pptx file? Default = `FALSE`

`pptxName` A character string, the name of the .pptx file to which the plot is exported. Ignored if exportPptx = `FALSE`.

References

Millet et al. (2016) Genome-wide analysis of yield in Europe: Allelic effects vary with drought and heat scenarios. Plant Physiology, October 2016, Vol. 172, p. 749–764

Segura et al. (2012) An efficient multi-locus mixed-model approach for genome-wide association studies in structured populations. Nature Genetics, June 2012, Vol. 44, p. 825–830.

Examples

```
## Create a gData object Using the data from the DROPS project.
## See the included vignette for a more extensive description on the steps.
data(dropsMarkers)
data(dropsMap)
data(dropsPheno)

## Add genotypes as row names of dropsMarkers and drop Ind column.
rownames(dropsMarkers) <- dropsMarkers[["Ind"]]
dropsMarkers <- dropsMarkers[colnames(dropsMarkers) != "Ind"]

## Add genotypes as row names of dropsMap.
rownames(dropsMap) <- dropsMap[["SNP.names"]]

## Rename Chomosome and Position columns.
colnames(dropsMap)[match(c("Chromosome", "Position"),
                  colnames(dropsMap))] <- c("chr", "pos")

## Convert phenotypic data to a list.
dropsPhenoList <- split(x = dropsPheno, f = dropsPheno[["Experiment"]])

## Rename Variety_ID to genotype and select relevant columns.
dropsPhenoList <- lapply(X = dropsPhenoList, FUN = function(trial) {
  colnames(trial)[colnames(trial) == "Variety_ID"] <- "genotype"
  trial <- trial[c("genotype", "grain.yield", "grain.number", "seed.size",
                "anthesis", "silking", "plant.height", "tassel.height",
```

```
                  "ear.height")]
return(trial)
})

## Create gData object.
gDataDrops <- createGData(geno = dropsMarkers, map = dropsMap,
                          pheno = dropsPhenoList)

## Run single trait GWAS for trait 'grain.yield' for trial Mur13W.

GWASDrops <- runSingleTraitGwas(gData = gDataDrops,
                                trials = "Mur13W",
                                traits = "grain.yield")

## Create a manhattan plot.
plot(GWASDrops)

## Manually set a threshold for significant snps and add a title.
plot(GWASDrops,
    yThr = 3.5,
    title = "Manhattan plot for Mur13W")

## Restrict plot to part of chr 6.
plot(GWASDrops,
    yThr = 3.5,
    chr = 6,
    startPos = 0,
    endPos = 110000000)

## Create a qq plot.
plot(GWASDrops,
    plotType = "qq",
    title = "QQ plot for Mur13W")

## Create a QTL plot.
plot(GWASDrops,
    plotType = "qtl",
    title = "QTL plot for Mur13W")

## Manually set a threshold and don't show vertical lines.
plot(GWASDrops,
    plotType = "qtl",
    yThr = 3.5,
    printVertGrid = FALSE,
    title = "QTL plot for Mur13W")
```

---

readPLINK *Read PLINK binary data*

Description

Read PLINK binary data and save in gData format. This is a wrapper around snpStats::read.plink in the Bioconductor package `snpStats`. This package needs to be installed for the function to work.

Usage

```
readPLINK(bed, bim, fam, ...)
```

Arguments

`bed`	The name of the file containing the packed binary SNP genotype data. It should have the extension .bed; If it doesn't, then this extension will be appended.
`bim`	The file containing the SNP descriptions. If not specified `bed` is used with its file extension replaced by bim.
`fam`	The file containing subject (and, possibly, family) identifiers. This is basically a tab-delimited "pedfile". If not specified `bed` is used with its file extension replaced by fam.
`...`	Further arguments passed to snpStats::read.plink.

Value

An object of class `gData`.

`readVcf` *Read variant call format data*

Description

Read variant call format (VCF) data and save in gData format. This is a wrapper around vcfR::read.vcfR in the package `vcfR`. This package needs to be installed for the function to work.

Usage

```
readVcf(vcfFile, ...)
```

Arguments

`vcfFile`	The name of the vcf file. This can either be a plain text file with extension (.vcf), or a gzipped file with extencion (.vcf.gz).
`...`	Further arguments passed to vcfR::read.vcfR.

Value

An object of class `gData`.

References

Knaus BJ, Grünwald NJ (2017). "VCFR: a package to manipulate and visualizevariant call format data in R." *Molecular Ecology Resources*, *17*(1), 44-53. ISSN 757, doi:10.1111/17550998.12549.

`runSingleTraitGwas` *Perform single-trait GWAS*

Description

`runSingleTraitGwas` performs a single-trait Genome Wide Association Study (GWAS) on phenotypic and genotypic data contained in a `gData` object. A covariance matrix is computed using the EMMA algorithm (Kang et al., 2008), the Newton-Raphson algorithm (Tunnicliffe, 1989) in the `sommer` package, or the modified Henderson algorithm in the `LMMsolver` package. Then a Generalized Least Squares (GLS) method is used for estimating the marker effects and corresponding p-values. This is done using either one kinship matrix for all chromosomes or different chromosome-specific kinship matrices for each chromosome. Significant SNPs are selected based on a user defined threshold.

Usage

```
runSingleTraitGwas(
  gData,
  traits = NULL,
  trials = NULL,
  covar = NULL,
  snpCov = NULL,
  kin = NULL,
  kinshipMethod = c("astle", "IBS", "vanRaden"),
  remlAlgo = c("EMMA", "NR", "Hend"),
  GLSMethod = c("single", "multi"),
  useMAF = TRUE,
  MAF = 0.01,
  MAC = 10,
  genomicControl = FALSE,
  thrType = c("bonf", "fixed", "small", "fdr"),
  alpha = 0.05,
  LODThr = 4,
  nSnpLOD = 10,
  pThr = 0.05,
  rho = 0.5,
  sizeInclRegion = 0,
  minR2 = 0.5,
  nCores = NULL
)
```

Arguments

gData	An object of class gData containing at least map, markers and pheno.
traits	A vector of traits on which to run GWAS. These can be either numeric indices or character names of columns in pheno. If NULL, GWAS is run on all traits.
trials	A vector of trials on which to run GWAS. These can be either numeric indices or character names of list items in pheno. If NULL, GWAS is run for all trials. GWAS is run for the selected trials in sequential order.
covar	An optional vector of covariates taken into account when running GWAS. These can be either numeric indices or character names of columns in covar in gData. If NULL no covariates are used.
snpCov	An optional character vector of snps to be included as covariates.
kin	An optional kinship matrix or list of kinship matrices. These matrices can be from the matrix class as defined in the base package or from the dsyMatrix class, the class of symmetric matrices in the Matrix package. If GLSMethod = "single" then one matrix should be provided, if GLSMethod = "multi", a list of chromosome specific matrices of length equal to the number of chromosomes in map in gData. If NULL then matrix kinship in gData is used. If both kin is provided and gData contains a matrix kinship then kin is used.
kinshipMethod	An optional character indicating the method used for calculating the kinship matrix(ces). Currently "astle" (Astle and Balding, 2009), "IBS" and "vanRaden" (VanRaden, 2008) are supported. If a kinship matrix is supplied either in gData or in parameter kin, kinshipMethod is ignored.
remlAlgo	A character string indicating the algorithm used to estimate the variance components. Either EMMA, for the EMMA algorithm, NR, for the Newton-Raphson algorithm (using sommer::mmes), or Hend for the modified Henderson algorithm (using LMMsolver::LMMsolve).
GLSMethod	A character string indicating the method used to estimate the marker effects. Either single for using a single kinship matrix, or multi for using chromosome specific kinship matrices.
useMAF	Should the minor allele frequency be used for selecting SNPs for the analysis. If FALSE, the minor allele count is used instead.
MAF	The minor allele frequency (MAF) threshold used in GWAS. A numerical value between 0 and 1. SNPs with MAF below this value are not taken into account in the analysis, i.e. p-values and effect sizes are put to missing (NA). Ignored if useMAF is FALSE.
MAC	A numerical value. SNPs with minor allele count below this value are not taken into account for the analysis, i.e. p-values and effect sizes are set to missing (NA). Ignored if useMAF is TRUE.
genomicControl	Should genomic control correction as in Devlin and Roeder (1999) be applied?
thrType	A character string indicating the type of threshold used for the selection of candidate loci. See Details.
alpha	A numerical value used for calculating the LOD-threshold for thrType = "bonf" and the significant p-Values for thrType = "fdr".

LODThr	A numerical value used as a LOD-threshold when thrType = "fixed".
nSnpLOD	A numerical value indicating the number of SNPs with the smallest p-values that are selected when thrType = "small".
pThr	A numerical value just as the cut off value for p-Values for thrType = "fdr".
rho	A numerical value used a the minimum value for SNPs to be considered correlated when using thrType = "fdr".
sizeInclRegion	An integer. Should the results for SNPs close to significant SNPs be included? If so, the size of the region in centimorgan or base pairs. Otherwise 0.
minR2	A numerical value between 0 and 1. Restricts the SNPs included in the region close to significant SNPs to only those SNPs that are in sufficient Linkage Disequilibrium (LD) with the significant snp, where LD is measured in terms of R^2. If for example sizeInclRegion = 200000 and minR2 = 0.5, then for every significant SNP also those SNPs whose LD (R^2) with the significant SNP is at least 0.5 AND which are at most 200000 away from this significant snp are included. Ignored if sizeInclRegion = 0.
nCores	A numerical value indicating the number of cores to be used by the parallel part of the algorithm. If NULL the number of cores used will be equal to the number of cores available on the machine - 1.

Value

An object of class GWAS.

Threshold types for the selection of candidate loci

For the selection of candidate loci from the GWAS output four different methods can be used. The method used can be specified in the function parameter thrType. Further parameters can be used to fine tune the method.

bonf The Bonferroni threshold, a LOD-threshold of $-log10(alpha/m)$, where m is the number of SNPs and alpha can be specified by the function parameter alpha

fixed A fixed LOD-threshold, specified by the function parameter LODThr

small The n SNPs with with the smallest p-Values are selected. n can be specified in nSnpLOD

fdr Following the algorithm proposed by Brzyski D. et al. (2017), SNPs are selected in such a way that the False Discovery Rate (fdr) is minimized. To do this, first the SNPs are restricted to the SNPs with a p-Value below pThr. Then clusters of SNPs are created using a two step iterative process in which first the SNP with the lowest p-Value is selected as cluster representative. This SNP and all SNPs that have a correlation with this SNP of ρ or higher will form a cluster. ρ can be specified as an argument in the function and has a default value of 0.5, which is a recommended starting value in practice. The selected SNPs are removed from the list and the procedure repeated until no SNPs are left. Finally to determine the number of significant clusters the first cluster is determined for which the p-Value of the cluster representative is not smaller than $cluster_{number} * \alpha/m$ where m is the number of SNPs and alpha can be specified by the function parameter alpha. All previous clusters are selected as significant.

References

Astle, William, and David J. Balding. 2009. Population Structure and Cryptic Relatedness in Genetic Association Studies. Statistical Science 24 (4): 451–71. doi:10.1214/09sts307.

Brzyski D. et al. (2017) Controlling the Rate of GWAS False Discoveries. Genetics 205 (1): 61-75. doi:10.1534/genetics.116.193987

Devlin, B., and Kathryn Roeder. 1999. Genomic Control for Association Studies. Biometrics 55 (4): 997–1004. doi:10.1111/j.0006341x.1999.00997.x.

Kang et al. (2008) Efficient Control of Population Structure in Model Organism Association Mapping. Genetics 178 (3): 1709–23. doi:10.1534/genetics.107.080101.

Millet, E. J., Pommier, C., et al. (2019). A multi-site experiment in a network of European fields for assessing the maize yield response to environmental scenarios - Data set. doi:10.15454/IASSTN

Rincent et al. (2014) Recovering power in association mapping panels with variable levels of linkage disequilibrium. Genetics 197 (1): 375–87. doi:10.1534/genetics.113.159731.

Segura et al. (2012) An efficient multi-locus mixed-model approach for genome-wide association studies in structured populations. Nature Genetics 44 (7): 825–30. doi:10.1038/ng.2314.

Sun et al. (2010) Variation explained in mixed-model association mapping. Heredity 105 (4): 333–40. doi:10.1038/hdy.2010.11.

Tunnicliffe W. (1989) On the use of marginal likelihood in time series model estimation. JRSS 51 (1): 15–27.

VanRaden P.M. (2008) Efficient methods to compute genomic predictions. Journal of Dairy Science 91 (11): 4414–23. doi:10.3168/jds.20070980.

See Also

summary.GWAS, plot.GWAS

Examples

```
## Create a gData object Using the data from the DROPS project.
## See the included vignette for a more extensive description on the steps.
data(dropsMarkers)
data(dropsMap)
data(dropsPheno)

## Add genotypes as row names of dropsMarkers and drop Ind column.
rownames(dropsMarkers) <- dropsMarkers[["Ind"]]
dropsMarkers <- dropsMarkers[colnames(dropsMarkers) != "Ind"]

## Add genotypes as row names of dropsMap.
rownames(dropsMap) <- dropsMap[["SNP.names"]]

## Rename Chomosome and Position columns.
colnames(dropsMap)[match(c("Chromosome", "Position"),
                  colnames(dropsMap))] <- c("chr", "pos")

## Convert phenotypic data to a list.
dropsPhenoList <- split(x = dropsPheno, f = dropsPheno[["Experiment"]])
```

```
## Rename Variety_ID to genotype and select relevant columns.
dropsPhenoList <- lapply(X = dropsPhenoList, FUN = function(trial) {
  colnames(trial)[colnames(trial) == "Variety_ID"] <- "genotype"
  trial <- trial[c("genotype", "grain.yield", "grain.number", "seed.size",
                "anthesis", "silking", "plant.height", "tassel.height",
                "ear.height")]
return(trial)
})

## Create gData object.
gDataDrops <- createGData(geno = dropsMarkers, map = dropsMap,
                          pheno = dropsPhenoList)

## Run single trait GWAS for trait 'grain.yield' for trial Mur13W.

GWASDrops <- runSingleTraitGwas(gData = gDataDrops,
                                trials = "Mur13W",
                                traits = "grain.yield")


## Run single trait GWAS for trait 'grain.yield' for trial Mur13W.
## Use chromosome specific kinship matrices calculated using vanRaden method.

GWASDropsMult <- runSingleTraitGwas(gData = gDataDrops,
                                    trials = "Mur13W",
                                    traits = "grain.yield",
                                    kinshipMethod = "vanRaden",
                                    GLSMethod = "multi")
```

---

summary.gData *Summary function for the class* gData

Description

Gives a summary for an object of S3 class gData.

Usage

```
## S3 method for class 'gData'
summary(object, ..., trials = NULL)
```

Arguments

object	An object of class gData.
...	Not used.
trials	A vector of trials to include in the summary. These can be either numeric indices or character names of list items in pheno. If NULL, all trials are included.

Value

A list with a most four components:

mapSum A list with number of markers and number of chromosomes in the map.

markerSum A list with number of markers, number of genotypes and the distribution of the values within the markers.

phenoSum A list of data.frames, one per trial with a summary of all traits within the trial.

covarSum A list of data.frames, one per trial with a summary of all covariates within the trial.

All components are only present in the output if the corresponding content is present in the gData object.

`summary.GWAS` *Summary function for the class* `GWAS`

Description

Gives a summary for an object of S3 class `GWAS`.

Usage

```
## S3 method for class 'GWAS'
summary(object, ..., trials = NULL, traits = NULL)
```

Arguments

`object`	An object of class `GWAS`.
`...`	Not used.
`trials`	A vector of strings or numeric indices indicating for which trials the summary should be made. If `NULL`, a summary is made for all trials.
`traits`	A vector of strings indicating which traits to include in the summary. If `NULL`, all traits are included in the summary.

Index

∗ **datasets**
dropsData, 5

codeMarkers, 2
createGData *(*gData*)*, 7

dropsData, 5
dropsMap *(*dropsData*)*, 5
dropsMarkers *(*dropsData*)*, 5
dropsPheno *(*dropsData*)*, 5

gData, 7

kinship, 9

LMMsolver::LMMsolve, *18*

plot.gData, 10
plot.GWAS, 12, *20*

readPLINK, 15
readVcf, 16
runSingleTraitGwas, 17

snpStats::read.plink, *16*
sommer::mmes, *18*
summary.gData, *8*, 21
summary.GWAS, *20*, 22

vcfR::read.vcfR, *16*