Package 'solarius'

September 19, 2018

Type Package

Title An R Interface to SOLAR

Version 0.3.0.2

Date 2015-12-12

Maintainer Andrey Ziyatdinov <andrey.ziyatdinov@upc.edu>

Description SOLAR is the standard software program to perform linkage and association mappings of the quantitative trait loci (QTLs) in pedigrees of arbitrary size and complexity. This package allows the user to exploit the variance component methods implemented in SOLAR. It automates such routine operations as formatting pedigree and phenotype data. It also parses the model output and contains summary and plotting functions for exploration of the results. In addition, solarius enables parallel computing of the linkage and association analyses, that makes the calculation of genome-wide scans more efficient. See <http://solar.txbiomedgenetics.org/> for more information about SOLAR.

License GPL (>= 3)

URL https://github.com/ugcd/solarius

LazyData false

OS_type unix

Imports methods, plyr (>= 1.8.1), ggplot2, data.table

Suggests tools, kinship2, scales, Matrix, gdata, doParallel, iterators, parallel, qqman, rsnps, grid, gridExtra

Collate 'package.R' 'datasets.R' 'data.lib.R' 'support.lib.R' 'solar.lib.R' 'assoc.lib.R' 'classSolarAssoc.R' 'classSolarPolygenic.R' 'plot.lib.R' 'polygenic.lib.R' 'solarAssoc.R' 'solarPolyAssoc.R' 'solarPolygenic.R' 'files.lib.R' 'classSolarMultipoint.R' 'solarMultipoint.R' 'multipoint.lib.R' 'transforms.lib.R' 'solarMultivar.R' 'solarBayesAvg.R' 'solarPar.R' 'modelPar.R'

RoxygenNote 5.0.1

NeedsCompilation no

Author Andrey Ziyatdinov [cre, aut],
Helena Brunel [aut],
Angel Martinez-Perez [aut],
Alfonso Buil [aut],

Alexandre Perera [cph],
Jose Manuel Soria [cph]

Repository CRAN

Date/Publication 2015-12-13 16:51:09

R topics documented:

annotate *Annotate SNPs in association study*

Description

The function calls annotateSNPs function, which does the job.

Usage

```
annotate(x, mode = c("significant", "top", "all"), alpha = 0.05,
  num.top = 10, ...)
```

Arguments

x	An object of class solarAssoc or a character vector of SNPs.
mode	A character with the mode of SNPs selection. Possible values are "significant", "top" and "all". The default value is "significant".
alpha	A numeric value from 0 to 1, the significance level after Bonferroni multiple-test correction. Corresponds to mode equal to "significant".
num.top	An integer value, the number of top SNPs to be annotated. Corresponds to mode equal to "top". The default value is 10.
...	Additional arguments passed to annotateSNPs.

Value

A data table with annotation results.

See Also

annotateSNPs

annotateSNPs *Annotate SNPs*

Description

The function annotates SNPs based on NCBI2R R package, in particular AnnotateSNPList function.

Usage

```
annotateSNPs(x, mode = c("significant", "top", "all"), alpha = 0.05,
  num.top = 10, query.size = 500, verbose = 0)
```

Arguments

x	An object of class solarAssoc or a character vector of SNPs.
mode	A character with the mode of SNPs selection. Possible values are "significant", "top" and "all". The default value is "significant".
alpha	A numeric value from 0 to 1, the significance level after Bonferroni multiple-test correction. Corresponds to mode equal to "significant".
num.top	An integer value, the number of top SNPs to be annotated. Corresponds to mode equal to "top". The default value is 10.
query.size	An integer, the maximum number of SNPs allowed for a single query to the NCBI database. The default value is 500. See also the help page for NCBI_snp_query function in rsnps package. If the number of SNPs is greater than query.size, then the query is split into batches automatically.
verbose	An integer, the verbose level. The default value is 0.
...	Additional arguments.

Details

See https://ncbi2r.wordpress.com/ for more details.

Value

A data table with annotation results.

---

availableTransforms *Get a list of available transforms.*

Description

Get a list of available transforms.

Usage

```
availableTransforms()
```

Value

A character vector of transform names.

Examples

```
availableTransforms()
```

`dat30` *dat30 data set adapted from multic R package*

Description

29 first families were selected from the complete data set of 12000 individuals. For a resulted subset of 174 individuals, a hundred of synthetic SNPs were randomly generated. Annotation information also was generated, mainly in order to plot the association results with Manhattan plot.

Format

(Phenotypes) A data frame `dat30` with 174 rows and 10 variables:

famid Family ID (29 unique ids).

id Individual ID.

fa Father ID.

mo Mother ID.

sex Individual gender (1 - male, 2 - female).

affect Affected status (1 - unaffected, 2 - affected).

class Class label.

trait1 Simulated phenotype 1.

trait2 Simulated phenotype 2.

age Age.

(Genotypes as covariates) A matrix `genocovdat30` with 174 rows and 100 columns. Row names are IDs of individuals, column names are names of SNPs.

(Annotation) A data frame `mapdat30` with 100 rows and 4 variables:

SNP SNP name.

chr Chromosome.

pos Position in bp.

gene Gene.

Details

Two simulated phenotypes possess a high genetic correlation.

Source

https://cran.r-project.org/package=multic

Examples

```
data(dat30)

str(dat30)

plotPed(dat30, 2) # plot the pedigree tree for family #2

## Not run:
kin2 <- solarKinship2(dat30)
plotKinship2(kin2)
plotKinship2(kin2[1:30, 1:30])


## End(Not run)
str(genocovdat30)

genocovdat30[1:5, 1:5]
str(mapdat30)

head(mapdat30)
```

`df2solar` *Export phenotype and pedigree data into SOLAR files*

Description

The function exports phenotype and pedigree data from R to SOLAR.

Usage

```
df2solar(df, dir, kinship, kin2.gz = "kin2.gz", sort.ped = TRUE)
```

Arguments

`df`	A data frame that containts both phenotype and pedigree data.
`dir`	A character with path to a directory, where SOLAR files to be created.
`kinship`	(optional) A kinship matrix to be exported.
`kin2.gz`	A character with file name of SOLAR file to store the kinship matrix.
`sort.ped`	Logical, indicating where pedigree IDs (FAM or FAMID) to be sorted. The default value is TRUE.

Details

The function (1) puts the data `df` in SOLAR format, (2) separates it into two parts, pedigree and phenotypes, and then (3) expots both data sets in the directory `dir`.

Pedigree or ID variables are detected by `matchIdNames` function.

explainedVarProp *Get explained variances for a group of SOLAR models.*

Description

Get explained variances for a group of SOLAR models.

Usage

```
explainedVarProp(mod)
```

Arguments

mod	An object of solarPolygenic class. See solarPolygenicClass.

Value

A data frame with two columns covariate and explainedVarProp.

getFormulaStr *Get formula string from a solarPolygenic object*

Description

The function returns a character string with formula. The formula is derived based on traits and covlist slots of input object.

Usage

```
getFormulaStr(x)
```

Arguments

x	An object of solarPolygenic object.

Value

A character string with formula.

loadExamplesPhen *Load the example data (GAW10) from SOLAR.*

Description

Function loads the example data distributed with SOLAR. See the SOLAR tutorial for description of the data set.

Usage

```
loadExamplesPhen(dat.dir)
```

Arguments

dat.dir	(optional) A character with path to a directory, where SOLAR files to be exported. If this argument is missing, a temporary directory is used.

Value

A data frame returned by `readPhen` function called with phen and ped files of the data set.

loadMulticPhen *Load the complete data set from R package multic*

Description

The function loads the complete data of 12,000 individuals, which is stored in .phen and .ped files. These two files were generated within R package `multic` and re-distributed in R package `solarius` (`extdata/solarOutput` directory).

Usage

```
loadMulticPhen()
```

Details

Function `readPhen` is used to read .phen and .ped files.

Examples

```
dat <- loadMulticPhen()
dim(dat)

data(dat30)
dim(dat30)
```

matchIdNames *Match ID column names*

Description

The function automates the process of renaming ID fields in data frame of phenotypes into a SOLAR format.

Usage

```
matchIdNames(fields, sex.optional = FALSE, skip.sex = FALSE)
```

Arguments

fields	The name of fields or colulm names in data frame, which are candidates to be ID fields.
sex.optional	Logical, indicating if through an error in the case SEX field is not found in input fields.
skip.sex	Logical, indicating if the search for SEX filds is completely skipped.

Value

A named character vector, that can be directly passed to rename function of plyr package.

Examples

```
# @ http://helix.nih.gov/Documentation/solar-6.6.2-doc/91.appendix_1_text.html#load
#solar> help file-pedigree
#
# The pedigree file consists of one record for each individual in the data
# set.  Each record must include the following fields:
#
#    ego ID, father ID, mother ID, sex
#
# In addition, a family ID is required when ego IDs are not unique across
# the entire data set.  If the data set contains genetically identical
# individuals, an MZ-twin ID must be present (as described below).  If an
# analysis of household effects is planned, a household ID can be included
# (also described below).
#
# The default field names are ID, FA, MO, SEX, FAMID, MZTWIN, and HHID.
#

fields <- c("id", "ID", "ids",
  "famid", "FAMID", "famidity",
  "mo", "MO", "mother", "MOTHER", "MOtrait", "motherland",
  "fa", "FA", "father", "FATHER", "fatherland",
  "sex", "SEX", "sexo")

### ID
# pass: id, ID
# filter: ids
grep("^id$|^ID$", fields, value = TRUE)
```

```
### FAMID
grep("^famid$|^FAMID$", fields, value = TRUE)

### MO
grep("^mo$|^MO$|^mother$|^MOTHER$", fields, value = TRUE)

### FA
grep("^fa$|^FA$|^father$|^FATHER$", fields, value = TRUE)

### SEX
grep("^sex$|^SEX$", fields, value = TRUE)
```

`matchMapNames` *Match map column names*

Description

The function searches for fields that correspond to those in SOLAR format of marker map file.

Usage

```
matchMapNames(fields)
```

Arguments

`fields`	The name of fields or colulm names in data frame, which are candidates to be map fields.

Value

A named character vector, that can be directly passed to `rename` function of `plyr` package.

Examples

```
data(dat50)
 matchMapNames(names(snpdata))
```

`modelPar` *Get a parameter value from* `solarius` *models.*

Description

Get a parameter value from `solarius` models.

Usage

```
modelPar(mod, par, ...)

modelParCPUtime(mod, format = "sec", ...)

modelParCores(mod)

modelParNumBatches(mod)
```

Arguments

mod	An object of solarPolygenic, solarMultipoint or solarAssoc classes. See solarPolygenicClass, solarMultipointClass and solarAssocClass.
par	A character, the parameter name.
...	Additional arguments.
format	A character, the format of the time value. The default value is "sec". The second possible value is "POSIX". This argument is only for modelParCPUtime function.

Value

A value of the given parameter.

---

package.file *Alternative to system.file*

Description

The function works as system.file, but takes care when the package is a local folder.

Usage

```
package.file(...)
```

Arguments

...	arguments to be passed to system.file

Details

The use case is when some data file or direcotry is needed to be loaded, and it is placed in inst/ directory of the package.

Value

Path returned by system.file.

Examples

```
mibddir <- package.file("extdata", "solarOutput", "solarMibds", package = "solarius")
mibddir

list.files(mibddir)
```

phenodata *dat50 data set adapted from FFBSKAT R package*

Description

A mixture of unrelated and related individuals was originally simulated in FFBSKAT R package to test methods of the variant-collapsing approach. 50 synthetic SNPs were generated for the association study. A custom kinship matrix is used to express the relationships among individuals. This data set is used here to test the ability of SOLAR to work with a custom kinship matrix in both polygenic and association analyses.

Format

(Phenotypes) A data frame `phenodata` with 66 rows and 4 variables:

id Individual ID.

sex Individual gender (0 - male, 1 - female).

age Age.

trait Simulated phenotype.

(Kinship) A square matrix `kin` with 66 rows and 66 columns.

(Genotypes) A matrix `genodata` with 66 rows and 50 columns.

(Genotypes as covariates) A matrix `genocovdata` with 66 rows and 50 columns.

(Annotation) A data frame `snpdata` with 100 rows and 4 variables:

name SNP name.

chrom Chromosome.

position Position in bp.

gene Gene.

Details

The genotypes are coded in the format such as 1/1, 1/2 and 2/2.

In addition to the original data set from FFBSKAT R package, a matrix of covariates was derived from the genotype data according to the additive model.

Source

http://mga.bionet.nsc.ru/soft/FFBSKAT/

Examples

```
data(dat50)

str(phenodata)
plotKinship2(2*kin)
str(genodata)

genodata[1:5, 1:5]
str(genocovdata)
```

```
genocovdata[1:5, 1:5]

# compare with the genotypes
genodata[1:5, 1:5]
str(snpdata)

head(snpdata)
```

`plotKinship2` *Plot the double kinship matrix*

Description

The main function that calls `imageKinship2` or `histKinship2` depending on value of `y` argument.

Usage

```
plotKinship2(x, y = c("image", "hist"))

imageKinship2(x)

histKinship2(x)
```

Arguments

`x`	A square matrix of double kinship coefficients.
`y`	A character, the type of the plot. Possible values are `"image"` and `"hist"`. The default value is `"image"`.

Details

`imageKinship2` function calls `image` function from `Matrix` package.

`histKinship2` function plots a histogram based on `ggplot2` package.

Examples

```
# load `kin` kinship matrix from `dat50` data set
data(dat50)
kin2 <- 2* kin # double kinship matrix

plotKinship2(kin2) # equivalent to `imageKinship2(kin2)`

plotKinship2(kin2, "hist") # equivalent to `histKinship2(kin2)`
```

plotManh *Plot the association results*

Description

Two Manhattan and quantile-quantile (QQ) plots are standard to explore the results from association studies.

Usage

```
plotManh(A, alpha = 0.05, main, ...)

plotQQ(A, df = 1, main, ...)
```

Arguments

A	An object of class solarAssoc.
alpha	A numeric value from 0 to 1, the p-value cut-off for Bonferroni multiple-test correction. The default value is 0.05.
main	A character string, main argument (title) to a plot function. If the argument is missing, the title contains information about the model formula and the number of SNPs used for plotting.
...	additional argument to a plot function.
df	An integer value, the degree of freedom. The default value is 1.

Details

plotManh function produces the Manhattan plot based on qqman package. The two red and blue lines, default in the original manhattan function of qqman package, are preserved. An additional black dashed line is added, that depicts the significance level according to Bonferroni multiple-test correction with alpha argument.

plotQQ function produces the QQ plot based on the same qqman package.

See Also

solarAssocClass

Examples

```
## Not run:
data(dat50)

assoc <- solarAssoc(trait ~ 1, phenodata,
 snpdata = genodata, snpmap = snpdata, kinship = kin)

plotManh(assoc) # equivalent to plot(assoc)

plotQQ(assoc) # equivalent to plot(assoc, "qq")


## End(Not run)
```

plotPed *Plot the pedigree*

Description

Plot the pedigree tree based on the ID-fields given in phenotypes. Plotting is based on `kinship2` package.

Usage

```
plotPed(data, ped)
```

Arguments

`data`	A data.frame of phenotypes.
`ped`	An integer or a character value, that indicates the pedigree.

Details

The ID fields are extracted from the data via `matchIdNames` function. The required fields are ID, FA, MO, SEX and FAMID.

See Also

`matchIdNames`

Examples

```
data(dat30)
plotPed(dat30, 1)
```

plotRes *Plot the residuals of a polygenic model*

Description

Plot the residuals on scatter or quantile-quantile plots.

Usage

```
plotRes(x, labels = FALSE, text.size = 4, ...)

plotResQQ(x, distribution = "norm", ..., line.estimate = NULL, conf = 0.9,
  labels = FALSE, text.size = 4)
```

Arguments

x	An object of class solarPolygenic.
labels	A logical value for plotRes function, indicating if the labels of IDs (which residuals are outside the 3 * sd interval) are to be plotted. A logical value for plotResQQ function, indicating if the samples (their IDs) outside the confidence intervals are to be plotted.
text.size	An integer, the text size of labels.
...	additional arguments.
distribution	A character, name of distribution of the residuals. The default value is "norm".
line.estimate	Function for estimation of QQ-line.
conf	A numeric value between 0 and 1, that represents the confidence boundary.

Details

plotRes function makes a scatter plot of fitted values vs. residuals. Note that the residuals returned by SOLAR include both random effects, i.e. house-hold, genetic and residuals itself.

plotResQQ function plots quantile-quantile (QQ) plot of the residuals.

See Also

solarPolygenicClass

Examples

```
## Not run:
### basic (univariate) polygenic model
mod <- solarPolygenic(trait1 ~ age + sex, dat30)

plotRes(mod)

plotResQQ(mod)


## End(Not run)
```

---

readPhen *Read plain-text table files phen and ped*

Description

The function reads the phenotype and pedigree data in SOLAR format, i.e. plain-text tables.

Usage

```
readPhen(phen.file, sep.phen = ",", ped.file, sep.ped = ",",
  header = TRUE, stringsAsFactors = FALSE, id.unique = TRUE, sex.optional)
```

Arguments

phen.file	A character, path to phen file.
sep.phen	A character, the field separator in phen file. The default value is ",".
ped.file	(optional) A character, path to ped file.
sep.ped	A character, the field separator in ped file. The default value is ",".
header	Logical, indicating whether the file contains the names of the variables as its first line. The default value is TRUE.
stringsAsFactors	logical, indicating whether character vectors to be converted to factors. The default value is FALSE.
id.unique	logical, indicating whether the IDs of individuals must be unique. The default value is TRUE.
sex.optional	logical, indicating whether the SEX field must be presented. The default value is TRUE if ped.file is specified, and it is FALSE otherwise.

Value

A data frame of phenotype data merged from two phen and ped files (merged by ID filed).

---

snpdata2solar *Export snp genotypes, genotype covariates and amp to SOLAR*

Description

A list of functions allows to pass SNPs data from R to SOLAR.

Usage

```
snpdata2solar(mat, dir)

snpcovdata2solar(mat, dir, nGTypes = FALSE, out)

snpmap2solar(map, dir)
```

Arguments

mat	A matrix of genotypes or genotypes as covariates to be exported.
dir	A character with path where SOLAR files to be created.
nGTypes	Logical, whether a column nGTypes to be added to snp.genocov file. The default value is FALSE.
out	(optional) A list, that contains the names for snp.genocov and snp.geno-list files. This argument is internally used in solarAssoc function.
map	A data frame with annotation (map) information for SNPs.

Details

snpdata2solar function (1) exports the data set of genotypes stored in mat itto SOLAR files, (2) runs solar command 'snp load solar.gen' to check the data is loaded ok; (3) if two output files were not created, throws an error.

snpcovdata2solar function emulates the 'snp load' SOLAR command. Two output files are produced: 'snp.genocov' and 'snp.geno-list'. The steps are the following: (1) add prefix 'snp_' to SNP names; (2) (optional) compute stats on # genotyped individuals (columns 'nGTypes'); (3) write data and metadata into files.

snpmap2solar function (1) separates data by chromosome; (2) write the table into SOLAR map file; (3) check if the map file is OK by running SOLAR command load map -basepair <filename>

Note

In association analysis (soalrAssoc function) the step of loading SNP maps is skipped. It seems that SOLAR does not use this information when doing association.

Examples

```
# Example of `snp.genocov` file:
# id,nGTypes,snp_s1,snp_s2,...
# 1,50,0,0,...
# 2,50,0,0,...

# Example of `snp.geno-list` file:
# snp_s1
# snp_s2
# ...
```

---

solar *Call SOLAR program from R*

Description

The function calls SOLAR via system function, that invokes the OS command (which is solar) specified by the command argument. SOLAR is required to be installed in the OS.

Usage

```
solar(cmd, dir, result = TRUE, ignore.stdout = TRUE,
  ignore.stderr = FALSE, ...)
```

Arguments

cmd	A vector of characters, that contains the commands to be passed to SOLAR.
dir	A character, path to the directory, where SOLAR-related files (phenotypes, pedigree, markers) were previously created.
result	A logical, intern argument to be passed to system function. The default value is TRUE.
ignore.stdout	A logical, ignore.stdout argument to be passed to system function. The default value is FALSE.

ignore.stderr	A logical, ignore.stderr argument to be passed to system function. The default value is TRUE.
...	additional arguments (which are not used).

Details

This is the core function in the interface between R and SOLAR.

---

solarAssoc *Run association analysis.*

Description

The association analysis is conducted in the following sequence: parse input files of SNP markers, export data to a directory by df2solar function, run the polygenic analysis in a directory, run the association analysis on the top of the polygenic analysis, parse output files and store results in an object of solarAssoc class (see solarAssocClass).

Usage

```
solarAssoc(formula, data, dir, kinship, traits, covlist = "1", snpformat,
  snpdata, snpcovdata, snpmap, snplist, snpind, genocov.files, snplists.files,
  snpmap.files, mga.files, plink.ped, plink.map, plink.raw,
  assoc.outformat = c("df", "outfile", "outfile.gz"), assoc.outdir,
  assoc.options = "", cores = getOption("cores"), batch.size = 1000, ...,
  verbose = 0)
```

Arguments

formula	an object of class formula or one that can be coerced to that class. It is a symbolic description of fixed effects (covariates) to be fitted.
data	A data frame containing the variables in the model, including ID fields needed to construct random effects: genetic and house-hold (both optional). Other classes such as list, environment or object coercible by as.data.frame to a data frame are not supported.
dir	an optional character string, the name of directory, where SOLAR performs the analysis. In this case, the analysis within related input/output files is conducted in the given folder instead of a temporary one (the default work flow).
kinship	A matrix of the kinship coefficients (custom kinship matrix). The IDs are required to be in row and column names.
traits	a vector of characters to specify trait(s) in the model. It is alternative to the formula interface.
covlist	a vector of characters to specify fixed effects (covariates) in the model. It is alternative to the formula interface. The default value is "1".
snpformat	a character, the format of SNP data passsed by snpdata argument. Currently, this argument is not used.
snpdata	A matrix of SNP data. SNPs are given in the columns, and individuals correspond to the rows. The IDs of individuals are required to be in row and column names.

snpcovdata	A matrix of SNP data, which are converted to covariates (numeric format). SNPs are given in the columns, and individuals correspond to the rows. The IDs of individuals are required to be in row and column names.
snpmap	A data.frame of annotation for SNPs.
snplist	a vector of characters, the names of SNPs to be used in the analysis. This argument may be used when a subset of SNPs is of the interest.
snpind	a vector of positive integers, the indices of SNPs to be used in the analysis. This argument may be used when a subset of SNPs is of the interest.
genocov.files	A vector of characters, the file paths to genocov SOLAR files.
snplists.files	A vector of characters, the file paths to snplists SOLAR files.
snpmap.files	A vector of characters (optional), the file paths to snpmap SOLAR files.
mga.files	A list with 2-3 elements, where each element is a vector of characters. This argument is an alternative to the other three genocov.files, snplists.files and snpmap.files. The element 3 of the list is optional.
plink.ped	A character, the file path to genotype data in plink .ped format. Two columns are used per genotype.
plink.map	A character, the file path to genotype annotation data in plink .map format.
plink.raw	A character, the file path to genotype data in allele-dosage plink format (an example plink command: plink --noweb --file dat50 --recodeA). One column is used per genotype.
assoc.outformat	A character, the output format. Possible values are "df", "outfile" and "outfile.gz". Currently, the only supported output format is "df". That means the table of results is stored in snpf slot of a returned object.
assoc.outdir	a character, the path to the output directory. Currently, this argument is not used.
assoc.options	A character, specific options to be passed to mga SOLAR command.
cores	A positive integer, the number of cores for parallel computing. The default value is taken from getOption("cores"). If the default value is NULL then the number of cores is 1.
batch.size	An integer, the number of SNPs per batch for parallel computation. The default value is 1000.
...	Arguments to be passed to solarPolygenic function. For example, one of such arguments may be polygenic.settings = "option EnableDiscrete 0". Arguments of solarMultipoint, which are also passed to solarPolygenic, include formula, data, dir, kinship, traits and covlist.
verbose	An non-negative integer of the verbose level. The default value is 0.

Note

solarAssoc function accepts input genetic data in three formats: SOLAR (genocov.files, snplists.files, snpmap.files and param mga.files arguments), R data frame or matrix (snpdata, snpcovdata and snpmap arguments), and plink (plink.ped, plink.map and plink.raw arguments).

For large-size problems, the user is recommended to prepare the genetic data in SOLAR format and to split them into batches of size, for example, 1,000 markers. The use of the other two R and plink formats is not optimized for large-scale scenarios.

Examples

```
### load data
data(dat50)
dim(phenodata)
dim(kin)
dim(genodata)

## Not run:
### basic (univariate) association model with a custom kinship
mod <- solarAssoc(trait~age+sex, phenodata,
  kinship = kin, snpdata = genodata)
mod$snpf # table of results for 50 SNPs

## End(Not run)
```

`solarAssocClass` *S3 class solarAssoc.*

Description

S3 class solarAssoc.

Usage

```
## S3 method for class 'solarAssoc'
print(x, ...)

## S3 method for class 'solarAssoc'
plot(x, y = "manh", ...)

## S3 method for class 'solarAssoc'
summary(object, alpha = 0.05, ...)
```

Arguments

`x`	An object of class `solarAssoc`.
`...`	Additional arguments.
`y`	Character argument for `plot` method, indicating the type of plot. The default value is `"manh"`.
`object`	An object of class `solarAssoc`.
`alpha`	Numeric argument between 0 and 1 for `summary` method, indicating the significance level after Bonferroni multiple-test correction. The default value is 0.05.

solarBaeysAvgPolygenicClass *S3 class solarBaeysAvgPolygenic*

Description

S3 class solarBaeysAvgPolygenic

Usage

```
## S3 method for class 'solarBaeysAvgPolygenic'
print(x, ...)

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

Arguments

x	An object of class solarBaeysAvgPolygenic.
...	Additional arguments.
object	An object of class solarBaeysAvgPolygenic.

solarKinship2 *Compute empirical double kinship matrix by SOLAR*

Description

The function runs SOLAR to evaluate the double kinship matirx of the given data.

Usage

```
solarKinship2(df, dir, ...)
```

Arguments

df	A data frame that containts both phenotype and pedigree data. To be passed to df2solar function.
dir	(optional) A character with path to a directory, where SOLAR files to be created. If this argument is missing, a temporary folder is created and further removed, when the job is done. To be passed to df2solar function.
...	Additional arguments to be passed to df2solar.

Details

The function (1) puts the data df in SOLAR format, (2) separates it into two parts, pedigree and phenotypes, and then (3) expots both data sets in the directory dir, (4) read the specific 'phi2.gz' file, where the kinship coefficients multiplied by 2 are stored by SOLAR.

Note

IDs in df are assumed to be not duplicated.

solarMultipoint *Run multipoint linkage analysis.*

Description

The linkage analysis is conducted in the following sequence: parse input files of multipoint IBD matrices, export data to a directory by df2solar function, run the polygenic analysis in a directory, run the linkage analysis on the top of the polygenic analysis, parse output files and store results in an object of solarMultipoint class (see solarMultipointClass).

Usage

```
solarMultipoint(formula, data, dir, kinship, traits, covlist = "1", mibddir,
  chr, interval, multipoint.options = "", multipoint.settings = "",
  cores = getOption("cores"), ..., verbose = 0)
```

Arguments

formula	an object of class formula or one that can be coerced to that class. It is a symbolic description of fixed effects (covariates) to be fitted.
data	A data frame containing the variables in the model, including ID fields needed to construct random effects: genetic and house-hold (both optional). Other classes such as list, environment or object coercible by as.data.frame to a data frame are not supported.
dir	an optional character string, the name of directory, where SOLAR performs the analysis. In this case, the analysis within related input/output files is conducted in the given folder instead of a temporary one (the default work flow).
kinship	A matrix of the kinship coefficients (custom kinship matrix). The IDs are required to be in row and column names.
traits	a vector of characters to specify trait(s) in the model. It is alternative to the formula interface.
covlist	a vector of characters to specify fixed effects (covariates) in the model. It is alternative to the formula interface. The default value is "1".
mibddir	A character, the name of directory with files representing the IBD matrices in SOLAR format. These matrices can be evaluated by SOLAR or by other program. See SOLAR help for ibd command for more details (http://solar.txbiomedgenetics.org/doc/91.appendix_1_text.html#ibd).
chr	A character or one that can be coerced to that class, the value to be passed to SOLAR command chromosome. If it is missing, the default value is "all". See SOLAR help for chromosome command for more details (http://solar.txbiomedgenetics.org/doc/91.appendix_1_text.html#chromosome).
interval	A character or one that can be coerced to that class, the value to be passed to SOLAR command interval. If it is missing, the default value is 1. See SOLAR help for interval command for more details.
multipoint.options	A character of options to be passed to multipoint SOLAR command. For example, one or more LOD scores may be specified to run multi-pass linkage analysis. If the argument is "3", that means a two-pass linkage scan with condition of the highest LOD of 3 to start the second pass. See SOLAR help page for

multipoint command for more details (http://solar.txbiomedgenetics.org/doc/91.appendix_1_text.html#multipoint). The default value is "".

multipoint.settings
: A vector of characters, that contains SOLAR commands to be executed just before calling multipoint. For example, the fine mapping for the linkage scan can be disabled by setting the given argument to "finemap off". The default value is "".

cores
: A positive integer, the number of cores for parallel computing. The default value is taken from getOption("cores"). If the default value is NULL then the number of cores is 1.

...
: Arguments to be passed to solarPolygenic function. For example, one of such arguments may be polygenic.settings = "option EnableDiscrete 0". Arguments of solarMultipoint, which are also passed to solarPolygenic, include formula, data, dir, kinship, traits and covlist.

verbose
: An non-negative integer of the verbose level. The default value is 0.

Note

The user is responsible for computation of the MIBD matrices required by solarMultipoint function via mibddir argument. Estimation of the IBD matrices is out of the scope of solarius package. More information is available in http://solar.txbiomedgenetics.org/doc/05.chapter.html.

Examples

```
### load phenotype data
data(dat30)

### load marker data
mibddir <- system.file('extdata', 'solarOutput',
  'solarMibdsCsv', package = 'solarius')
list.files(mibddir)

## Not run:
### basic (univariate) linkage model
mod <- solarMultipoint(trait1 ~ 1, dat30,
  mibddir = mibddir, chr = 5)
mod$lodf # table of results (LOD scores) (the highest 3.56)

### basic (bivariate) linkage model
mod <- solarMultipoint(trait1 + trait2 ~ 1, dat30,
  mibddir = mibddir, chr = 5)
mod$lodf # table of results (LOD scores) (the highest 2.74)

### two-pass linkage model
mod <- solarMultipoint(trait1 ~ 1, dat30,
  mibddir = mibddir, chr = 5,
  multipoint.options = "3")
mod$lodf # table of results (LOD scores, 1 pass) (the highest 2.74)
mod$lodf2 # table of results (LOD scores, 2 pass) (all nearly zero LOD scores)


## End(Not run)
```

solarMultipointClass *S3 class solarMultipoint.*

Description

S3 class solarMultipoint.

Usage

```
## S3 method for class 'solarMultipoint'
print(x, ...)

## S3 method for class 'solarMultipoint'
plot(x, pass = 1, main, xlab, faceting = TRUE,
  ...)

## S3 method for class 'solarMultipoint'
summary(object, ...)

## S3 method for class 'solarMultipoint'
tabplot(object, LOD.thr = 1.5, plot.null = TRUE,
  ...)
```

Arguments

x	An object of class solarMultipoint.
...	Additional arguments.
pass	Integer argument for plot method, indicating whether which pass in multi-passs linkage scan to be plotted. The default value is 1.
main	An argument for plot method.
xlab	An argument for plot method.
faceting	An argument for plot method. The default value is TRUE.
object	An object of class solarMultipoint.
LOD.thr	An argument for tabplot method. The default value is 1.5.
plot.null	An argument for tabplot method. The default value is TRUE.

solarMultivarClass *S3 class solarMultivar.*

Description

S3 class solarMultivar.

Usage

```
## S3 method for class 'solarMultivar'
print(x, ...)
```

Arguments

x	An object of class solarMultivar.
...	Additional arguments.

---

solarPar *Get a parameter value from SOLAR model files.*

Description

Get a parameter value from SOLAR model files.

Usage

```
solarPar(mod, par)

solarParIndividuals(mod)

solarParH2rP(mod)

solarParKurtosis(mod)

solarParCovlistP(mod)

solarParCovlistChi(mod)

solarParRhoP(mod)

solarParRhoPSE(mod)

solarParRhoPP(mod)

solarParRhoPOK(mod)

solarParPvar(mod, modelname = "null0.mod")
```

Arguments

mod	An object of solarPolygenic class. See solarPolygenicClass.
par	A character, the parameter name.
modelname	A character, the file name of a model. The default value is "null0.mod". This argument is only for solarParPvar function.

Value

A value of the given parameter.

solarPolyAssoc *S3 class solarPolyAssoc.*

Description

S3 class solarPolyAssoc.

Usage

```
## S3 method for class 'solarPolyAssoc'
print(x, ...)
```

Arguments

x	An object of class solarPolyAssoc.
...	Additional arguments.

solarPolygenic *Run polygenic analysis.*

Description

The polygenic analysis is conducted in the following sequence: export data to a directory by df2solar function, form a SOLAR call with a list of settings and options, execute SOLAR by solar function, parse output files and store results in an object of solarPolygenic class (see solarPolygenicClass).

Usage

```
solarPolygenic(formula, data, dir, kinship, traits, covlist = "1",
  covtest = FALSE, screen = FALSE, household = as.logical(NA),
  transforms = character(0), alpha = 0.05, polygenic.settings = "",
  polygenic.options = "", verbose = 0, ...)
```

Arguments

formula	an object of class formula or one that can be coerced to that class. It is a symbolic description of fixed effects (covariates) to be fitted. If the model does not have any covariates, then the formula looks like trait ~ 1, where 1 means the trait mean parameter.
data	A data frame containing the variables in the model, including ID fields needed to construct random effects: genetic and house-hold (both optional). Other classes such as list, environment or object coercible by as.data.frame to a data frame are not supported.
dir	an optional character string, the name of directory, where SOLAR performs the analysis. In this case, the analysis within related input/output files is conducted in the given folder instead of a temporary one (the default work flow).

`kinship` A matrix of the kinship coefficients (custom kinship matrix). The IDs are required to be in row and column names. Currently, it does not work for unrelated individuals (SOLAR issue).

`traits` a vector of characters to specify trait(s) in the model. It is alternative to the formula interface.

`covlist` a vector of characters to specify fixed effects (covariates) in the model. It is alternative to the formula interface. It could be convenient to indicate covariates in the SOLAR format, for example, `"age^1,2,3#sex"` that means `sex age agesex age^2 age^2sex ag` The default value is `"1"`.

`covtest` a logical value, indicating whether to test the significance of the fixed effects (covariates). Likelihood ratio test (LRT) is used by SOLAR. `polygenic` SOLAR command is called with a combination of `-screen -all` options. As a result, `cf` slot will have p-values in `pval` column. The default value is `FALSE`.

`screen` a logical value, indicating whether to screen the fixed effects (covariates). `polygenic` SOLAR command is called with `-screen` option. As a result, only significant covariates will be maintained in the model. The default value is `FALSE`.

`household` a logical value, saying to **forcedly** include or exclude the house-hold effect. The default value is `as.logical(NA)`, that means the following behavior in SOLAR. If `data` has `hhid` or similar column, then the house-hold effect is added to the model and tested by SOLAR. Otherwise, there is no any variable indicating the house-hold effect neither in `data` nor in the model. If `household` is `TRUE`, then `polygenic` SOLAR command is called with `-keephouse` option. If `household` is `FALSE`, then `house` SOLAR command is not called previously to calling `polygenic` SOLAR command (modeling of the house-hold effect is omitted).

`transforms` a named vector of characters, indicating the transformations to be applied to traits. A list of available transforms is returned by function `availableTransforms`. If the model is univariate, the name of transformation is not necessary and can be omitted. The default value is `character(0)`.

`alpha` a number between 0 an 1, that is the value of `-prob` option of `polygenic` SOLAR command. That is the probability level for keeping covariates as significant. The default value in SOLAR is 0.1, but the default value here is `0.05`. This parameter makes the `polygenic` SOLAR call to be like `polygenic -prob 0.05`.

`polygenic.settings`
A vector of characters, that contains SOLAR commands to be executed just before calling `polygenic`. For example, the liability threshold model applied to a binary trait (the default behavior in SOLAR). This behavior is disabled by setting the given argument to `"option EnableDiscrete 0"`. The default value is `""`.

`polygenic.options`
A character of options to be passed to `polygenic` SOLAR command. For example, the comprehensive analysis of a bivariate model might be parametrized by setting this parameter to `"-testrhoe -testrhog -testrhoc -testrhop -rhopse"`. See SOLAR help page for `polygenic` command for more details (http://solar.txbiomedgenetics.org/doc/91.appendix_1_text.html#polygenic). The default value is `""`.

`verbose` An non-negative integer of the verbose level. The default value is `0`.

`...` additional parameters to be passed to other functions called inside of `solarPolygenic`. For example, it might be a parameter `log.base` for `transformTrait` function in the case `transform` is equal to `"log"`.

Value

An object of solarPolygenic class. See solarPolygenicClass.

Examples

```
### load data and check out ID names
data(dat30)
matchIdNames(names(dat30))

## Not run:
### basic (univariate) polygenic model
mod <- solarPolygenic(trait1 ~ age + sex, dat30)

### (univariate) polygenic model with parameters
mod <- solarPolygenic(trait1 ~ age + sex, dat30, covtest = TRUE)
mod$cf # look at test statistics for covariates

### basic (bivariate) polygenic model
mod <- solarPolygenic(trait1 + trait2 ~ 1, dat30)
mod$vcf # look at variance components

### (bivariate) polygenic model with trait specific covariates
mod <- solarPolygenic(trait1 + trait2 ~ age + sex(trait1), dat30)

### (bivariate) polygenic model with a test of the genetic correlation
mod <- solarPolygenic(trait1 + trait2 ~ 1, dat30, polygenic.options = "-testrhog")
mod$lf # look at a p-value of the test

### transforms for (univariate) polygenic model
mod <- mod <- solarPolygenic(trait1 ~ 1, dat30, transforms = "log")

### transforms for (bivariate) polygenic model
mod <- solarPolygenic(trait1 + trait2 ~ 1, dat30,
   transforms = c(trait1 = "log", trait2 = "inormal"))

### SOLAR format of introducing covariates
mod <- solarPolygenic(traits = "trait1", covlist = "age^1,2,3#sex", data =  dat30)
mod$cf # 8 covariate terms will be printed


## End(Not run)
```

---

solarPolygenicClass *S3 class solarPolygenic.*

Description

S3 class solarPolygenic.

Usage

```
## S3 method for class 'solarPolygenic'
print(x, ...)
```

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

## S3 method for class 'solarPolygenic'
residuals(object, trait = FALSE, ...)
```

Arguments

x	An object of class solarPolygenic.
...	Additional arguments.
object	An object of class solarPolygenic.
trait	Logical argument for residuals method, indicating whether values of trait to be returned instead of residuals. The default value is FALSE.

---

solarReadFiles *Read SOLAR output files in a directory*

Description

The function is used, for example, to read the files in the output directory of the polygenic model.

Usage

```
solarReadFiles(dir)
```

Arguments

dir	A character, path to the directory.

Value

A list with file contents acquired by readLines function.

---

tabplot *S3 method tabplot*

Description

S3 method tabplot

Usage

```
tabplot(object, ...)
```

Arguments

object	An object.
...	Additional arguments.

transformData *Apply transforms to a data set.*

Description

The function looks for traits in colulms of input data frame, call transformTrait function per trait, and rename the traits. By default, the name of a transformed trait is updated to one with a prefixed given in transform.prefix argument (the default value is "tr_"). Such renaming is assumed to make the user aware that the trait is transformed.

Usage

```
transformData(transforms, data, transform.prefix = "tr_", ...)
```

Arguments

transforms	A named character vector of transforms, where traits are given in the names of the vector.
data	A matrix or a data.frame, where the column names represent the names of traits.
transform.prefix	A character vector, that is a prefix to be added to the name of transformed trait. The default value is "tr_".
...	Additional parameters to be passed to transformTrait function called inside of transformData. For example, it might be a parameter log.base for transformTrait function in the case transform is equal to "log".

Details

This function is internally called in solarPolygenic if transforms argument is specified. In this case of the polygenic analysis, the transform operation is invisible to the user. However, it is recommended to manually transform traits in other linkage and association analyses.

Value

A matrix or a data.frame of the transformed data.

See Also

availableTransforms, transformTrait

transformTrait *Transform a trait.*

Description

Transform a trait.

Usage

```
transformTrait(x, transform, mult = 1, ...)
```

Arguments

x	a numeric vector (of a trait).
transform	a character vector, the name of transformation. Possible values are returned by availableTransforms function.
mult	A numeric, the multiplicator for the transformed value of a trait. The default value is 1.
...	additional parameters passed to internal transform_trait_* functions. Possible parameters might be log.base, log.intercept ("log" transformation).

Value

A numeric vector, which contains the transformed values (of a trait).

See Also

availableTransforms, transformData

Examples

```
library(plyr)
library(ggplot2)

data(dat30)
dat <- mutate(dat30,
   inormal_trait1 = transformTrait(trait1, "inormal"))

ggplot(dat, aes(trait1)) + geom_histogram()
ggplot(dat, aes(inormal_trait1)) + geom_histogram()
```

Index