Package ‘coloc’

May 17, 2019

Type Package

Imports data.table, ggplot2, snpStats, BMA, reshape, methods

Suggests knitr, testthat, bindata, rmarkdown

Title Colocalisation Tests of Two Genetic Traits

Version 3.2-1

Date 2019-05-16

Maintainer Chris Wallace <cew54@cam.ac.uk>

Description Performs the colocalisation tests described in
Plagnol et al (2009) <doi:10.1093/biostatistics/kxn039>,
Wallace et al (2013) <doi:10.1002/gepi.21765> and
Giambartolomei et al (2013) <doi:10.1371/journal.pgen.1004383>.

License GPL

LazyLoad yes

VignetteBuilder knitr

RoxygenNote 6.1.1

URL https://github.com/chr1swallace/coloc

BugReports https://github.com/chr1swallace/coloc/issues

NeedsCompilation no

Author Chris Wallace [aut, cre],
Claudia Giambartolomei [aut],
Vincent Plagnol [ctb]

Repository CRAN

Date/Publication 2019-05-17 10:10:06 UTC

R topics documented:

coloc-package 2
approx.bf.estimates 3
approx.bf.p 4

bf . . . 4
coloc-class . . . 5
coloc.abf . . . 6
coloc.abf.datasets . . . 7
coloc.abf.snpStats . . . 8
coloc.bma . . . 9
coloc.test . . . 11
coloc.test.summary . . . 13
colocABF-class . . . 15
colocPCs-class . . . 15
combine.abf . . . 16
eta . . . 17
fillin . . . 18
finemap.abf . . . 18
logdiff . . . 19
logsum . . . 20
pcs.model . . . 21
pcs.prepare . . . 22
plot . . . 24
process.dataset . . . 25
sdY.est . . . 26
Var.data . . . 26
Var.data.cc . . . 27

Index **28**

`coloc-package` *Colocalisation tests of two genetic traits*

Description

Performs the colocalisation tests described in Plagnol et al (2009) and Wallace et al (in preparation) and draws some plots.

Details

`coloc.test()` tests for colocalisation and returns an object of class `coloc`.

Author(s)

Chris Wallace <chris.wallace@cimr.cam.ac.uk>

References

Plagnol et al (2009). Statistical independence of the colocalized association signals for type 1 diabetes and RPS26 gene expression on chromosome 12q13. Biostatistics 10:327-34.

http://www.ncbi.nlm.nih.gov/pubmed/19039033

Wallace et al (2013). Statistical Testing of Shared Genetic Control for Potentially Related Traits. Genetic Epidemiology 37:802-813.

http://www.ncbi.nlm.nih.gov/pmc/articles/PMC4158901/

Giambartolomei et al (2014). Bayesian Test for Colocalisation between Pairs of Genetic Association Studies Using Summary Statistics. PLOS Genet e1004383.

http://www.ncbi.nlm.nih.gov/pmc/articles/PMC4022491/

`approx.bf.estimates` *Internal function, approx.bf.estimates*

Description

Internal function, approx.bf.estimates

Usage

```
approx.bf.estimates(z, V, type, suffix = NULL, sdY = 1)
```

Arguments

`z`	normal deviate associated with regression coefficient and its variance
`V`	its variance
`type`	"quant" or "cc"
`suffix`	suffix to append to column names of returned data.frame
`sdY`	standard deviation of the trait. If not supplied, will be estimated.

Details

Calculate approximate Bayes Factors using supplied variance of the regression coefficients

Value

data.frame containing lABF and intermediate calculations

Author(s)

Vincent Plagnol, Chris Wallace

`approx.bf.p` *Internal function, approx.bf.p*

Description

Internal function, approx.bf.p

Usage

```
approx.bf.p(p, f, type, N, s, suffix = NULL)
```

Arguments

`p`	p value
`f`	MAF
`type`	"quant" or "cc"
`N`	sample size
`s`	proportion of samples that are cases, ignored if type=="quant"
`suffix`	suffix to append to column names of returned data.frame

Details

Calculate approximate Bayes Factors

Value

data.frame containing lABF and intermediate calculations

Author(s)

Claudia Giambartolomei, Chris Wallace

`bf` *Bayes factors to compare specific values of eta*

Description

Summarise the evidence for/against specific values or ranges of eta using bayes factors

Usage

```
bf(object)

## S4 method for signature 'colocBayes'
bf(object)
```

Arguments

`object`	of class `colocBayes`

Details

Only available for `colocBayes` objects, and you need to specify the specific values of interest using the `bayes.factor` argument when doing the proportional coloc analysis

Value

a matrix of Bayes factors

Author(s)

Chris Wallace

`coloc-class` *Classes* `"coloc"` *and* `"colocBayes"`

Description

Classes designed to hold objects returned by function `coloc.test` which performs a test of the null hypothesis that two genetic traits colocalise - that they share a common causal variant.

Objects from the Class

Objects can be created by calls to the function `coloc.test()`. Class `colocBayes` extends class `coloc`.

Author(s)

Chris Wallace.

References

Wallace et al (2012). Statistical colocalisation of monocyte gene expression and genetic risk variants for type 1 diabetes. Hum Mol Genet 21:2815-2824. `http://europepmc.org/abstract/MED/22403184`

Plagnol et al (2009). Statistical independence of the colocalized association signals for type 1 diabetes and RPS26 gene expression on chromosome 12q13. Biostatistics 10:327-34. `http://www.ncbi.nlm.nih.gov/pubmed/19039033`

See Also

`coloc.test`, `coloc.test.summary`, `coloc.bma`

Examples

```
showClass("coloc")
showClass("colocBayes")
```

`coloc.abf` *Fully Bayesian colocalisation analysis using Bayes Factors*

Description

Bayesian colocalisation analysis

Usage

```
coloc.abf(dataset1, dataset2, MAF = NULL, p1 = 1e-04, p2 = 1e-04,
  p12 = 1e-05)
```

Arguments

`dataset1`	a list with the following elements **pvalues** P-values for each SNP in dataset 1 **N** Number of samples in dataset 1 **MAF** minor allele frequency of the variants **beta** regression coefficient for each SNP from dataset 1 **varbeta** variance of beta **type** the type of data in dataset 1 - either "quant" or "cc" to denote quantitative or case-control **s** for a case control dataset, the proportion of samples in dataset 1 that are cases **sdY** for a quantitative trait, the population standard deviation of the trait. if not given, it can be estimated from the vectors of varbeta and MAF **snp** a character vector of snp ids, optional. If present, it will be used to merge dataset1 and dataset2. Otherwise, the function assumes dataset1 and dataset2 contain results for the same SNPs in the same order. Some of these items may be missing, but you must give • always`type` • if `type`=="cc"`s` • if `type`=="quant" and `sdY` known`sdY` • if `type`=="quant" and `sdY` unknown`beta`, `varbeta`, `N`, `MAF` and then either • `pvalues`, `MAF` • `beta`, `varbeta`
`dataset2`	as above, for dataset 2
`MAF`	Common minor allele frequency vector to be used for both dataset1 and dataset2, a shorthand for supplying the same vector as parts of both datasets
`p1`	prior probability a SNP is associated with trait 1, default 1e-4
`p2`	prior probability a SNP is associated with trait 2, default 1e-4
`p12`	prior probability a SNP is associated with both traits, default 1e-5

Details

This function calculates posterior probabilities of different causal variant configurations under the assumption of a single causal variant for each trait.

If regression coefficients and variances are available, it calculates Bayes factors for association at each SNP. If only p values are available, it uses an approximation that depends on the SNP's MAF and ignores any uncertainty in imputation. Regression coefficients should be used if available.

Value

a list of two `data.frames`:

- summary is a vector giving the number of SNPs analysed, and the posterior probabilities of H0 (no causal variant), H1 (causal variant for trait 1 only), H2 (causal variant for trait 2 only), H3 (two distinct causal variants) and H4 (one common causal variant)
- results is an annotated version of the input data containing log Approximate Bayes Factors and intermediate calculations, and the posterior probability SNP.PP.H4 of the SNP being causal for the shared signal

Author(s)

Claudia Giambartolomei, Chris Wallace

`coloc.abf.datasets` *Bayesian colocalisation analysis using data.frames*

Description

Bayesian colocalisation analysis using data.frames

Usage

```
coloc.abf.datasets(df1, df2, snps = intersect(setdiff(colnames(df1),
  response1), setdiff(colnames(df2), response2)), response1 = "Y",
  response2 = "Y", ...)
```

Arguments

`df1`	dataset 1
`df2`	dataset 2
`snps`	col.names for snps
`response1`	col.name for response in dataset 1
`response2`	col.name for response in dataset 2
`...`	parameters passed to `coloc.abf.snpStats`

Details

Converts genetic data to snpStats objects, generates p values via score tests, then runs `coloc.abf`

Value

output of `coloc.abf`

Author(s)

Chris Wallace

`coloc.abf.snpStats` *Bayesian colocalisation analysis using snpStats objects*

Description

Bayesian colocalisation analysis using snpStats objects

Usage

```
coloc.abf.snpStats(X1, X2, Y1, Y2, snps = intersect(colnames(X1),
  colnames(X2)), type1 = c("quant", "cc"), type2 = c("quant", "cc"),
  s1 = NA, s2 = NA, ...)
```

Arguments

`X1`	genetic data for dataset 1
`X2`	genetic data for dataset 2
`Y1`	response for dataset 1
`Y2`	response for dataset 2
`snps`	optional subset of snps to use
`type1`	type of data in Y1, "quant" or "cc"
`type2`	type of data in Y2, "quant" or "cc"
`s1`	the proportion of samples in dataset 1 that are cases (only relevant for case control samples)
`s2`	the proportion of samples in dataset 2 that are cases (only relevant for case control samples)
`...`	parameters passed to `coloc.abf`

Details

Generates p values via score tests, then runs `coloc.abf`

Value

output of `coloc.abf`

Author(s)

Chris Wallace

`coloc.bma` *Wrapper to use colocalization testing within a Bayesian model averaging structure.*

Description

Performs the colocalisation tests described in Plagnol et al (2009) and Wallace et al (2012).

Usage

```
coloc.bma(df1, df2, snps = intersect(setdiff(colnames(df1), c(response1,
  stratum1)), setdiff(colnames(df2), c(response2, stratum2))),
  response1 = "Y", response2 = "Y", stratum1 = NULL,
  stratum2 = NULL, family1 = "binomial", family2 = "binomial",
  bayes = !is.null(bayes.factor), thr = 0.01, nsnps = 2,
  n.approx = 1001, bayes.factor = NULL, plot.coeff = FALSE,
  r2.trim = 0.95, quiet = FALSE, bma = FALSE, ...)
```

Arguments

`df1, df2`	Each is a dataframe, containing response and potential explanatory variables for two independent datasets.
`snps`	The SNPs to consider as potential explanatory variables
`response1, response2`	The names of the response variables in `df1` and `df2` respectively
`stratum1`	optional column name of df1 that gives stratum information
`stratum2`	optional column name of df2 that gives stratum information
`family1, family2`	the error family for use in `glm`
`bayes`	Logical, indicating whether to perform Bayesian inference for the coefficient of proportionality, eta. If `bayes.factor` is supplied, Bayes factors are additionally computed for the specificed values. This can add a little time as it requires numerical integration, so can be set to FALSE to save time in simulations, for example.
`thr`	posterior probability threshold used to trim SNP list. Only SNPs with a marginal posterior probability of inclusion greater than this with one or other trait will be included in the full BMA analysis

`nsnps`	number of SNPs required to model both traits. The BMA analysis will average over all possible `nsnp` SNP models, subject to `thr` above.
`n.approx`	number of values at which to numerically approximate the posterior
`bayes.factor`	if true, compare specific models
`plot.coeff`	deprecated
`r2.trim`	for pairs SNPs with r2>`r2.trim`, only one SNP will be retained. This avoids numerical instability problems caused by including two highly correlated SNPs in the model.
`quiet`	suppress messages about how the model spaced is trimmed for BMA
`bma`	if true (default), average over models
`...`	other parameters passed to `coloc.test`

Details

This is a test for proportionality of regression coefficients from two independent regressions. Analysis can either be based on a profile likelihood approach, where the proportionality coefficient, `eta`, is replaced by its maximum likelihood value, and inference is based on a chisquare test (`p.value`), or taking a hybrid-Bayesian approach and integrating the p value over the posterior distribution of `eta`, which gives a posterior predictive p value. The Bayesian approach can also be used to give a credible interval for `eta`. See the references below for further details.

Value

a `coloc` or `colocBayes` object

Author(s)

Chris Wallace

References

Wallace et al (2012). Statistical colocalisation of monocyte gene expression and genetic risk variants for type 1 diabetes. Hum Mol Genet 21:2815-2824. http://europepmc.org/abstract/MED/22403184

Plagnol et al (2009). Statistical independence of the colocalized association signals for type 1 diabetes and RPS26 gene expression on chromosome 12q13. Biostatistics 10:327-34. http://www.ncbi.nlm.nih.gov/pubmed/19039033

Examples

```
 ## simulate covariate matrix (X) and continuous response vector (Y)
 ## for two populations/triats Y1 and Y2 depend equally on f1 and f2
 ## within each population, although their distributions differ between
 ## populations.  They are compatible with a null hypothesis that they
 ## share a common causal variant
set.seed(1)
 X1 <- matrix(rbinom(2000,1,0.4),ncol=4)
 Y1 <- rnorm(500,rowSums(X1[,1:2]),2)
 X2 <- matrix(rbinom(2000,1,0.6),ncol=4)
 Y2 <- rnorm(500,rowSums(X2[,1:2]),5)

 boxplot(list(Y1,Y2),names=c("Y1","Y2"))

 ## fit and store linear model objects
 colnames(X1) <- colnames(X2) <- sprintf("f%s",1:ncol(X1))
 summary(lm1 <- lm(Y1~f1+f2+f3+f4,data=as.data.frame(X1)))
 summary(lm2 <- lm(Y2~f1+f2+f3+f4,data=as.data.frame(X2)))


## test colocalisation using bma
df1=as.data.frame(cbind(Y1=Y1,X1))
df2=as.data.frame(cbind(Y2=Y2,X2))

result <- coloc.bma( df1, df2, snps=colnames(X1), response1="Y1", response2="Y2",
family1="gaussian", family2="gaussian",
nsnps=2,bayes.factor=c(1,2,3))
result
plot(result)

## test colocalisation when one dataset contains a stratifying factor in column named "s"
df1$s <- rbinom(500,1,0.5)
result <- coloc.bma( df1, df2, snps=colnames(X1), response1="Y1", response2="Y2",
stratum1="s",
family1="gaussian", family2="gaussian",
nsnps=2,bayes.factor=c(1,2,3))
result
plot(result)
```

`coloc.test` *Function to do colocalisation tests of two traits*

Description

Performs the colocalisation tests described in Plagnol et al (2009) and Wallace et al (2012).

Usage

```
coloc.test(X, Y, vars.drop = NULL, ...)
```

Arguments

`X`	Either an lm or glm object for trait 1. The intersection of `names(coefficients(X))` and `names(coefficients(Y))` is used to identify SNPs in common which will be tested for colocalisation. Any Intercept term is dropped, but other covariates should have distinct names or be listed in `vars.drop` to avoid them being included in the colocalisation test.

`Y`	Either an lm or glm object for trait 2.
`vars.drop`	Character vector naming additional variables in either regression which are not SNPs and should not be used in the colocalisation test. They should appear in `c(names(coefficients(X)),names(coefficients(Y)))`
`...`	other arguments passed to `coloc.test.summary()`.

Details

This is a test for proportionality of regression coefficients from two independent regressions. Analysis can either be based on a profile likelihood approach, where the proportionality coefficient, `eta`, is replaced by its maximum likelihood value, and inference is based on a chisquare test (`p.value`), or taking a hybrid-Bayesian approach and integrating the p value over the posterior distribution of `eta`, which gives a posterior predictive p value. The Bayesian approach can also be used to give a credible interval for `eta`. See the references below for further details.

Value

a numeric vector with 3 named elements:

`eta.hat`	The estimated slope.
`chisquare`	The chisquared test statistic
`n`	The number of snps used in the test. If eta were known, this would be the degrees of freedom of the test. Because eta has been replaced by its ML estimate, Plagnol et al suggest we expect the degrees of freedom to be n-1, but this requires the likelihood to be well behaved which is not always the case. We prefer to consider the posterior predictive p value.
`ppp`	The posterior predictive p value

Note

Plagnol et al's original test was available in his R package `QTLMatch v0.8` which now appears unavailable. The numerically identical test, extended to allow for more than two SNPs, can be found in this package by looking at the chisquare statistic and the degrees of freedom given by `chisquare()` and `df()` respectively.

Author(s)

Chris Wallace

References

Wallace et al (2012). Statistical colocalisation of monocyte gene expression and genetic risk variants for type 1 diabetes. Hum Mol Genet 21:2815-2824. http://europepmc.org/abstract/MED/22403184

Plagnol et al (2009). Statistical independence of the colocalized association signals for type 1 diabetes and RPS26 gene expression on chromosome 12q13. Biostatistics 10:327-34. http://www.ncbi.nlm.nih.gov/pubmed/19039033

Examples

```
  ## simulate covariate matrix (X) and continuous response vector (Y)
  ## for two populations/triats Y1 and Y2 depend equally on f1 and f2
  ## within each population, although their distributions differ between
  ## populations.  They are compatible with a null hypothesis that they
  ## share a common causal variant
set.seed(1)
  X1 <- matrix(rbinom(1000,1,0.4),ncol=2)
  Y1 <- rnorm(500,apply(X1,1,sum),2)
  X2 <- matrix(rbinom(1000,1,0.6),ncol=2)
  Y2 <- rnorm(500,2*apply(X2,1,sum),5)

  boxplot(list(Y1,Y2),names=c("Y1","Y2"))

  ## fit and store linear model objects
  colnames(X1) <- colnames(X2) <- c("f1","f2")
  summary(lm1 <- lm(Y1~f1+f2,data=as.data.frame(X1)))
  summary(lm2 <- lm(Y2~f1+f2,data=as.data.frame(X2)))

  ## test whether the traits are compatible with colocalisation
  ### ppp should be large (>0.05, for example), indicating that they are.
  par(mfrow=c(2,2))
  obj <- coloc.test(lm1,lm2,
           plots.extra=list(x=c("eta","theta"),
                            y=c("lhood","lhood")))
  plot(obj)
```

`coloc.test.summary` *Colocalisation testing using regression coefficients*

Description

Colocalisation testing supplying only regression coefficients and their variance-covariants matrices

Usage

```
coloc.test.summary(b1, b2, V1, V2, k = 1, plot.coeff = FALSE,
  plots.extra = NULL, bayes = !is.null(bayes.factor),
  n.approx = 1001, level.ci = 0.95, bayes.factor = NULL,
  bma = FALSE)
```

Arguments

`b1`	regression coefficients for trait 1
`b2`	regression coefficients for trait 2
`V1`	variance-covariance matrix for trait 1

`V2`	variance-covariance matrix for trait 2
`k`	Theta has a Cauchy(0,k) prior. The default, k=1, is equivalent to a uniform (uninformative) prior. We have found varying k to have little effect on the results.
`plot.coeff`	DEPRECATED. Please `plot()` returned object instead. `TRUE` if you want to generate a plot showing the coefficients from the two regressions together with confidence regions.
`plots.extra`	list with 2 named elements, x and y, equal length character vectors containing the names of the quantities to be plotted on the x and y axes. `x` is generally a sequence of `theta` and `eta`, with `y` selected from `post.theta`, the posterior density of theta, `chisq`, the chi-square values of the test, and `lhood`, the likelihood function.
`bayes`	Logical, indicating whether to perform Bayesian inference for the coefficient of proportionality, eta. If `bayes.factor` is supplied, Bayes factors are additionally computed for the specificed values. This can add a little time as it requires numerical integration, so can be set to FALSE to save time in simulations, for example.
`level.ci, n.approx`	`level.ci` denotes the required level of the credible interval for `eta`. This is calculated numerically by approximating the posterior distribution at `n.approx` distinct values.
`bayes.factor`	Calculate Bayes Factors to compare specific values of eta. `bayes.factor` should either a numeric vector, giving single value(s) of `eta` or a list of numeric vectors, each of length two and specifying ranges of eta which should be compared to each other. Thus, the vector or list needs to have length at least two.
`bma`	parameter set to `TRUE` when `coloc.test` is called by `coloc.bma`. DO NOT SET THIS WHEN CALLING `coloc.test` DIRECTLY!

Details

Typically this should be called from `coloc.test()` or `coloc.bma()`, but is left as a public function, to use at your own risk, if you have some other way to define the SNPs under test.

Value

an object of class coloc, colocBayes or colocBMA

Author(s)

Chris Wallace

colocABF-class — *Class* "colocABF" *holds objects returned by the* coloc.abf *function*

Description

Objects can be created by calls to the function `coloc.abf()`.

Author(s)

Chris Wallace.

See Also

`coloc.abf`

Examples

```
showClass("colocABF")
```

colocPCs-class — *Class* "colocPCs"

Description

designed to hold objects returned by function `pcs.prepare` which generates a principal component summary of two genotype matrices in a form suitable for use in the function `pcs.model`.

designed to hold objects returned by function `pcs.prepare` which generates a principal component summary of two genotype matrices in a form suitable for use in the function `pcs.model`.

Objects from the Class

Objects can be created by calls to the function `pcs.prepare()`.

Objects can be created by calls to the function `pcs.prepare()`.

Author(s)

Chris Wallace.

Chris Wallace.

References

Wallace et al (2012). Statistical colocalisation of monocyte gene expression and genetic risk variants for type 1 diabetes. Hum Mol Genet 21:2815-2824. http://europepmc.org/abstract/MED/22403184

Plagnol et al (2009). Statistical independence of the colocalized association signals for type 1 diabetes and RPS26 gene expression on chromosome 12q13. Biostatistics 10:327-34. http://www.ncbi.nlm.nih.gov/pubmed/19039033

Wallace et al (2012). Statistical colocalisation of monocyte gene expression and genetic risk variants for type 1 diabetes. Hum Mol Genet 21:2815-2824. http://europepmc.org/abstract/MED/22403184

Plagnol et al (2009). Statistical independence of the colocalized association signals for type 1 diabetes and RPS26 gene expression on chromosome 12q13. Biostatistics 10:327-34. http://www.ncbi.nlm.nih.gov/pubmed/19039033

See Also

`pcs.prepare`, `pcs.model`

`pcs.prepare`, `pcs.model`

Examples

```
showClass("colocPCs")


showClass("colocPCs")
```

`combine.abf` *combine.abf*

Description

Internal function, calculate posterior probabilities for configurations, given logABFs for each SNP and prior probs

Usage

```
combine.abf(l1, l2, p1, p2, p12)
```

Arguments

`l1`	merged.df$lABF.df1
`l2`	merged.df$lABF.df2
`p1`	prior probability a SNP is associated with trait 1, default 1e-4
`p2`	prior probability a SNP is associated with trait 2, default 1e-4
`p12`	prior probability a SNP is associated with both traits, default 1e-5

Value

named numeric vector of posterior probabilities

Author(s)

Claudia Giambartolomei, Chris Wallace

`eta` *Methods to extract information from a* `coloc` *or* `colocBayes` *object*

Description

Extract information from a `coloc` object.

Usage

```
eta(object)
```

Arguments

`object`	Object returned by `coloc.test()` or `coloc.bma()` functions.

Details

`eta()` returns eta.hat, the maximum likelihood value of eta.

`theta()` returns theta.hat, the maximum likelihood value of eta.

`summary()` returns a summary, giving eta, chisquare statistic, number of SNPs/PCs, p value and, if a `colocBayes` object, the ppp.value

`ci()` returns the credible interval, or `NA` if not calculated.

Author(s)

Chris Wallace.

See Also

`coloc.test`, `pcs.prepare`

fillin — *Impute missing genotypes*

Description

Impute missing genotypes in a snpMatrix object in each SNP in turn, conditional on all the others.

Usage

```
fillin(X, bp = 1:ncol(X), strata = NULL)
```

Arguments

`X`	a snpMatrix object
`bp`	optional vector giving basepair positions of the SNPs
`strata`	optional vector giving stratificiation of the samples, one entry for each sample, and samples with the same value are assumed to come from a single strata

Value

a numeric matrix of imputed genotypes, 0,2 = homs, 1 = het

finemap.abf — *Bayesian finemapping analysis*

Description

Bayesian finemapping analysis

Usage

```
finemap.abf(dataset, p1 = 1e-04)
```

Arguments

`dataset` a list with the following elements

- **pvalues** P-values for each SNP in dataset 1
- **N** Number of samples in dataset 1
- **MAF** minor allele frequency of the variants
- **beta** regression coefficient for each SNP from dataset 1
- **varbeta** variance of beta
- **type** the type of data in dataset 1 - either "quant" or "cc" to denote quantitative or case-control
- **s** for a case control dataset, the proportion of samples in dataset 1 that are cases

	sdY for a quantitative trait, the population standard deviation of the trait. if not given, it can be estimated from the vectors of varbeta and MAF
	snp a character vector of snp ids, optional. If present, it will be used to merge dataset1 and dataset2. Otherwise, the function assumes dataset1 and dataset2 contain results for the same SNPs in the same order.
	Some of these items may be missing, but you must give • always`type` • if `type=="cc"`s • if `type=="quant"` and `sdY` known`sdY` • if `type=="quant"` and `sdY` unknown`beta`, `varbeta`, `N`, `MAF` and then either • `pvalues`, `MAF` • `beta`, `varbeta`
`p1`	prior probability a SNP is associated with the trait 1, default 1e-4

Details

This function calculates posterior probabilities of different causal variant for a single trait.

If regression coefficients and variances are available, it calculates Bayes factors for association at each SNP. If only p values are available, it uses an approximation that depends on the SNP's MAF and ignores any uncertainty in imputation. Regression coefficients should be used if available.

Value

a `data.frame`:

- an annotated version of the input data containing log Approximate Bayes Factors and intermediate calculations, and the posterior probability of the SNP being causal

Author(s)

Chris Wallace

`logdiff` *logdiff*

Description

Internal function, logdiff

Usage

```
logdiff(x, y)
```

Arguments

`x`	numeric
`y`	numeric

Details

This function calculates the log of the difference of the exponentiated logs taking out the max, i.e. insuring that the difference is not negative

Value

max(x) + log(exp(x - max(x,y)) - exp(y-max(x,y)))

Author(s)

Chris Wallace

`logsum` *logsum*

Description

Internal function, logsum

Usage

```
logsum(x)
```

Arguments

`x`	numeric vector

Details

This function calculates the log of the sum of the exponentiated logs taking out the max, i.e. insuring that the sum is not Inf

Value

max(x) + log(sum(exp(x - max(x))))

Author(s)

Claudia Giambartolomei

`pcs.model` *pcs.model*

Description

Functions to prepare principle component models for colocalisation testing

Usage

```
pcs.model(object, group, Y, stratum = NULL, threshold = 0.8,
  family = if (all(Y %in% c(0, 1))) {     "binomial" } else {
  "gaussian" })
```

Arguments

`object`	A colocPCs object, result of `pcs.prepare()`.
`group`	1 or 2, indicating which group of samples to extract from principal components matrix
`Y`	Numeric phenotype vector, length equal to the number of samples from the requested group
`stratum`	optional vector that gives stratum information
`threshold`	The minimum number of principal components which captures at least threshold proportion of the variance will be selected. Simulations suggest `threshold=0.8` is a good default value.
`family`	Passed to `glm()` function. `pcs.model` attempts to guess, either "binomial" if `Y` contains only 0s and 1s, "gaussian" otherwise.

Details

Prepares models of response based on principal components of two datasets for colocalisation testing.

Value

`pcs.prepare` returns a `colocPCs` object, `pcs.model` returns a `glm` object.

Author(s)

Chris Wallace

References

Wallace et al (2012). Statistical colocalisation of monocyte gene expression and genetic risk variants for type 1 diabetes. Hum Mol Genet 21:2815-2824. http://europepmc.org/abstract/MED/22403184

Plagnol et al (2009). Statistical independence of the colocalized association signals for type 1 diabetes and RPS26 gene expression on chromosome 12q13. Biostatistics 10:327-34. http://www.ncbi.nlm.nih.gov/pubmed/19039033

Examples

```
 ## simulate covariate matrix (X) and continuous response vector (Y)
 ## for two populations/triats Y1 and Y2 depend equally on f1 and f2
 ## within each population, although their distributions differ between
 ## populations.  They are compatible with a null hypothesis that they
 ## share a common causal variant, with the effect twice as strong for
 ## Y2 as Y1
set.seed(1)
 X1 <- matrix(rbinom(5000,1,0.4),ncol=10)
 Y1 <- rnorm(500,apply(X1[,1:2],1,sum),2)
 X2 <- matrix(rbinom(5000,1,0.6),ncol=10)
 Y2 <- rnorm(500,2*apply(X2[,1:2],1,sum),5)

 ## generate principal components object
 colnames(X1) <- colnames(X2) <- make.names(1:ncol(X1))
 pcs <- pcs.prepare(X1,X2)

 ## generate glm objects
 m1 <- pcs.model(pcs, group=1, Y=Y1)
 m2 <- pcs.model(pcs, group=2, Y=Y2)

 ## Alternatively, if one (or both) datasets have a known stratification, here simulated as
 S <- rbinom(500,1,0.5)
 ## specify this in pcs.model as
 m1 <- pcs.model(pcs, group=1, Y=Y1, stratum=S)

 ## test colocalisation using PCs
 coloc.test(m1,m2,plot.coeff=FALSE,bayes=FALSE)
```

`pcs.prepare` *Functions to prepare principle component models for colocalisation testing*

Description

Prepares principal components of two datasets for colocalisation testing.

Usage

```
pcs.prepare(X1, X2, impute = TRUE)
```

Arguments

`X1`	Either a SnpMatrix or numeric matrix of genetic data. Columns index SNPs, rows index samples.
`X2`	as X1
`impute`	if TRUE (default), impute missing genotypes

Details

If `X1` and `X2` are `SnpMatrix` objects, they are checked for missing data, and any missing values imputed by repeated use of `impute.snps` from the `snpStats` package.

Columns with common names are `rbind`ed together and principal components calculated using `prcomp`.

`pcs.model` can then be invoked to create `glm` objects.

Value

a `colocPCs` object.

Author(s)

Chris Wallace

References

Wallace et al (2012). Statistical colocalisation of monocyte gene expression and genetic risk variants for type 1 diabetes. Hum Mol Genet 21:2815-2824. `http://europepmc.org/abstract/MED/22403184`

Plagnol et al (2009). Statistical independence of the colocalized association signals for type 1 diabetes and RPS26 gene expression on chromosome 12q13. Biostatistics 10:327-34. `http://www.ncbi.nlm.nih.gov/pubmed/19039033`

Examples

```
 ## simulate covariate matrix (X) and continuous response vector (Y)
 ## for two populations/triats Y1 and Y2 depend equally on f1 and f2
 ## within each population, although their distributions differ between
 ## populations.  They are compatible with a null hypothesis that they
 ## share a common causal variant, with the effect twice as strong for
 ## Y2 as Y1
set.seed(1)
 X1 <- matrix(rbinom(5000,1,0.4),ncol=10)
 Y1 <- rnorm(500,apply(X1[,1:2],1,sum),2)
 X2 <- matrix(rbinom(5000,1,0.6),ncol=10)
 Y2 <- rnorm(500,2*apply(X2[,1:2],1,sum),5)
```

```
## generate principal components object
colnames(X1) <- colnames(X2) <- make.names(1:ncol(X1))
pcs <- pcs.prepare(X1,X2)

## generate glm objects
m1 <- pcs.model(pcs, group=1, Y=Y1)
m2 <- pcs.model(pcs, group=2, Y=Y2)

## test colocalisation using PCs
coloc.test(m1,m2,plot.coeff=FALSE,bayes=FALSE)
```

plot *Plotting functions for the coloc package*

Description

You can plot objects of class `coloc`, `colocBayes` and `colocABF`

Plot results of a coloc.abf run

Usage

```
plot(x, y, ...)

## S4 method for signature 'colocTWAS,missing'
plot(x)

## S4 method for signature 'coloc,missing'
plot(x, y, ...)

## S4 method for signature 'colocABF,missing'
plot(x, y, ...)

## S4 method for signature 'coloc,missing'
plot(x, y, ...)

## S4 method for signature 'colocPCs,missing'
plot(x)

abf.plot(coloc.obj, Pos = 1:nrow(coloc.obj@results), chr = NULL,
  pos.start = min(Pos), pos.end = max(Pos), trait1 = "trait 1",
  trait2 = "trait 2")
```

Arguments

x	object to be plotted
y	ignored

`...`	other arguments
`coloc.obj`	object of class `colocABF` returned by coloc.abf()
`Pos`	positions of all snps in ds1 or in ds2
`chr`	Chromosome
`pos.start`	lower bound of positions
`pos.end`	upper bound of positions
`trait1`	name of trait 1
`trait2`	name of trait 2

Details

If coloc.obj is missing, it will be created as coloc.obj=coloc.abf(ds1,ds2). Both ds1 and ds2 should contain the same snps in the same order

Value

no return value

a ggplot object

Author(s)

Hui Guo, Chris Wallace

`process.dataset` *process.dataset*

Description

Internal function, process each dataset list for coloc.abf

Usage

```
process.dataset(d, suffix)
```

Arguments

`d`	list
`suffix`	"df1" or "df2"

Value

data.frame with log(abf) or log(bf)

Author(s)

Chris Wallace

sdY.est *Estimate trait variance, internal function*

Description

Estimate trait standard deviation given vectors of variance of coefficients, MAF and sample size

Usage

```
sdY.est(vbeta, maf, n)
```

Arguments

vbeta	vector of variance of coefficients
maf	vector of MAF (same length as vbeta)
n	sample size

Details

Estimate is based on var(beta-hat) = var(Y) / (n * var(X)) var(X) = 2*maf*(1-maf) so we can estimate var(Y) by regressing n*var(X) against 1/var(beta)

Value

estimated standard deviation of Y

Author(s)

Chris Wallace

Var.data *Var.data*

Description

variance of MLE of beta for quantitative trait, assuming var(y)=1

Usage

```
Var.data(f, N)
```

Arguments

f	minor allele freq
N	sample number

Details

Internal function

Value

variance of MLE beta

Author(s)

Claudia Giambartolomei

`Var.data.cc` *Var.data*

Description

variance of MLE of beta for case-control

Usage

```
Var.data.cc(f, N, s)
```

Arguments

`f`	minor allele freq
`N`	sample number
`s`	???

Details

Internal function

Value

variance of MLE beta

Author(s)

Claudia Giambartolomei

Index

∗Topic **classes**
  coloc-class, 5
  colocABF-class, 15
  colocPCs-class, 15
∗Topic **methods**
  bf, 4
  eta, 17
∗Topic **package**
  coloc-package, 2

abf.plot *(plot)*, 24
approx.bf.estimates, 3
approx.bf.p, 4

bf, 4
bf,colocBayes-method *(bf)*, 4

ci *(eta)*, 17
ci,coloc-method *(eta)*, 17
ci,colocBayes-method *(eta)*, 17
coloc-class, 5
coloc-package, 2
coloc.abf, 6, *8*, *9*, *15*
coloc.abf.datasets, 7
coloc.abf.snpStats, *7*, 8
coloc.bma, *5*, 9, *14*
coloc.test, *5*, 11, *14*, *17*
coloc.test.summary, *5*, *12*, 13
colocABF-class, 15
colocBayes-class *(coloc-class)*, 5
colocBayesBMA *(coloc-class)*, 5
colocBMA *(coloc-class)*, 5
colocPCs-class, 15
combine.abf, 16

eta, 17
eta,coloc-method *(eta)*, 17
eta,colocBayes-method *(eta)*, 17

fillin, 18
finemap.abf, 18

logdiff, 19
logsum, 20

pcs.model, *15*, *16*, 21
pcs.prepare, *15–17*, 22
plot, 24
plot,coloc,missing-method *(plot)*, 24
plot,colocABF,missing-method *(plot)*, 24
plot,colocPCs,missing-method *(plot)*, 24
plot,colocTWAS,missing-method *(plot)*, 24
process.dataset, 25

sdY.est, 26
summary *(eta)*, 17
summary,coloc-method *(eta)*, 17
summary,colocBayes-method *(eta)*, 17

theta *(eta)*, 17
theta,coloc-method *(eta)*, 17
theta,colocBayes-method *(eta)*, 17

Var.data, 26
Var.data.cc, 27