Coloc: a package for colocalisation analyses

Chris Wallace

2013-05-22 Wed

Contents

1 A brief outline of colocalisation analysis **1**

2 Usage **2**

3 Proportional testing **2**
3.1 Principal components 2
3.2 Bayesian model averaging 4
3.3 Using Bayes Factors to compare specific values of η 7

4 (Approximate) Bayes Factor colocalisation analyses **8**
4.1 Introduction 8
4.2 The basic `coloc.abf` function 8
4.3 The wrapper function 9

5 The difference between proportional and ABF approaches **9**

1 A brief outline of colocalisation analysis

The coloc package can be used to perform genetic colocalisation analysis of two potentially related phenotypes, to ask whether they share common genetic causal variant(s) in a given region. There are a few key references which this vignette will not duplicate (see below), but, in brief, two approaches are implemented.

First, the proportional approach uses the fact that for two traits sharing causal variants, regression coefficients for either trait against any set of SNPs in the neighbourhood of those variants must be proportional. This test was first proposed by Plagnol et al. [1] in the context of evaluating whether expression of the gene *RPS26* mediated the association of type 1 diabetes to a region on chromosome 12q13 as had recently been proposed. The test addressed a common issue in genetics, and meant researchers could avoid the need to squint at parallel manhattan plots to decide whether two traits share causal variants. The function `coloc.test()` in this package evolved from code released by Vincent, but no longer available.

However, choosing **which** SNPs to use for the test is a problem. The obvious choice is to use those most strongly associated with one or other trait to maximise information, but doing so

[1] http://www.ncbi.nlm.nih.gov/pubmed/19039033

induces bias in the regression coefficients, which in turn leads to increased likelihood of spuriously rejecting the null of colocalisation, ie a quite substantially increased type 1 error rate [2]. I proposed two alternatives to address this problem, either using a principal component summary of genetic variation in the region to overcome the need to select a small set of test SNPs, implemented in `coloc.pcs()` and associated functions, or to use the ideas of Bayesian model averaging to average p values over SNP selections, generating posterior predictive p values, implemented in `coloc.bma()`.

Proportional testing, however, requires individual level genotype data, which are not always available. Claudia Giambartolomei and Vincent Plagnol have proposed an alternative method, which makes use of Jon Wakefield's work on determining approximate Bayes Factors from p values [3] to generate a Bayesian colocalisation analysis [4], implemented in the function `coloc.summaries()`. Note that it is possible to use Bayesian analysis for proportional testing too, in determining the posterior distribution of the proportionality constant η.

2 Usage

Let's simulate a small dataset, and compare the three methods.

```
Class "simdata" [in ".GlobalEnv"]

Slots:

Name:         df1        df2
Class: data.frame data.frame

[1] "show"

Generate 1 small sets of data
```

3 Proportional testing

3.1 Principal components

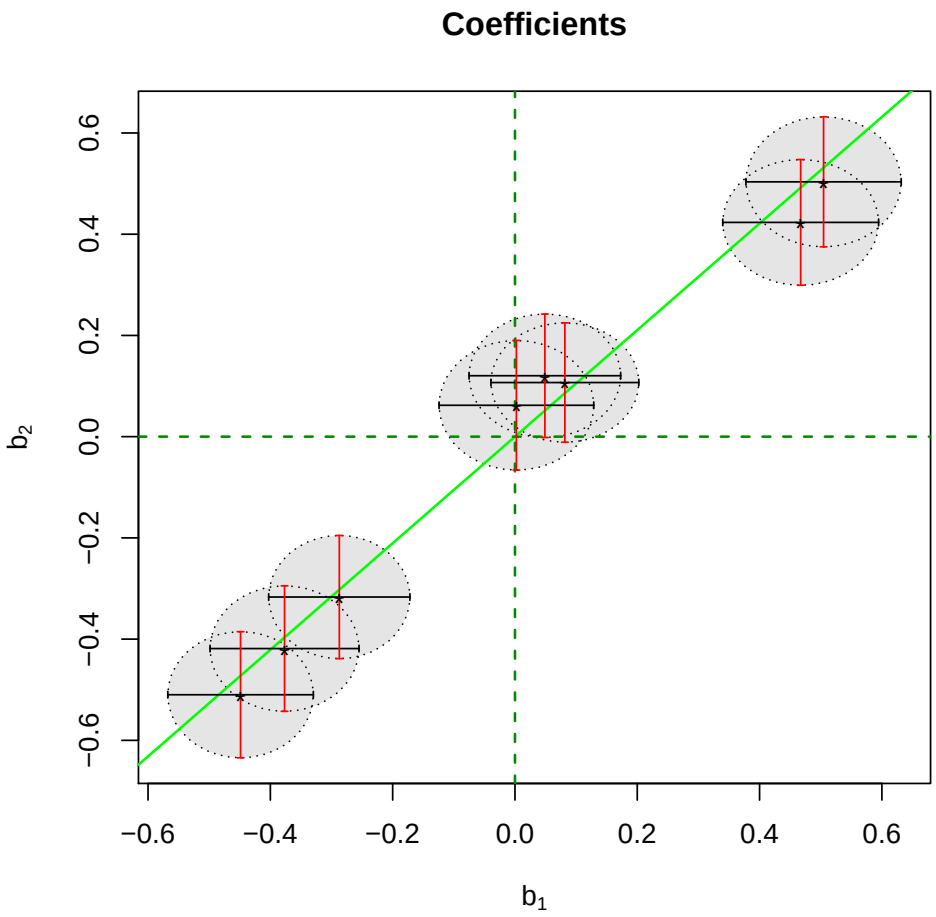
The code below first prepares a principal component object by combining the genotypes in the two dataset, then models the most informative components (the minimum set required to capture 80% of the genetic variation) in each dataset, before finally testing whether there is colocalisation between these models.

```
> ## run a coloc with pcs
> pcs <- pcs.prepare(data@df1[,-1], data@df2[,-1])
> pcs.1 <- pcs.model(pcs, group=1, Y=data@df1[,1], threshold=0.8)

selecting 8 components out of 10 to capture 0.8205794 of total variance.

> pcs.2 <- pcs.model(pcs, group=2, Y=data@df2[,1], threshold=0.8)
```

[2] http://arxiv.org/abs/1301.5510
[3] http://www.ncbi.nlm.nih.gov/pubmed/18642345
[4] http://arxiv.org/abs/1305.4022

```
selecting 8 components out of 10 to capture 0.8205794 of total variance.
```

```
> ct.pcs <- coloc.test(pcs.1,pcs.2)
```



The plot shows the estimated coefficients for each principal component modeled for traits 1 and 2 on the x and y axes, with circles showing the 95% confidence region. The points lie close to the line through the origin, which supports a hypothesis of colocalisation.

A little more information is stored in the `ct.pcs` object:

```
> ct.pcs
```

```
          eta.hat         chisquare                 n p.value.chisquare
        1.0528002         1.9060956         8.0000000         0.9648543
```

```
> str(summary(ct.pcs))
```

```
          eta.hat         chisquare                 n p.value.chisquare
        1.0528002         1.9060956         8.0000000         0.9648543
 Named num [1:4] 1.053 1.906 8 0.965
 - attr(*, "names")= chr [1:4] "eta.hat" "chisquare" "n" "p.value.chisquare"
```

The best estimate for the coefficient of proportionality, $\hat{\eta}$, is 1.13, and the null hypothesis of colocalisation is not rejected with a chisquare statistic of 5.27 based on 7 degrees of freedom ($n-1$ where the n is the number of components tested, and one degree of freedom was used in estimating η), giving a p value of 0.63. The `summary()` method returns a named vector of length 4 containing this information.

If more information is needed about η, then this is available if the `bayes` argument is supplied:

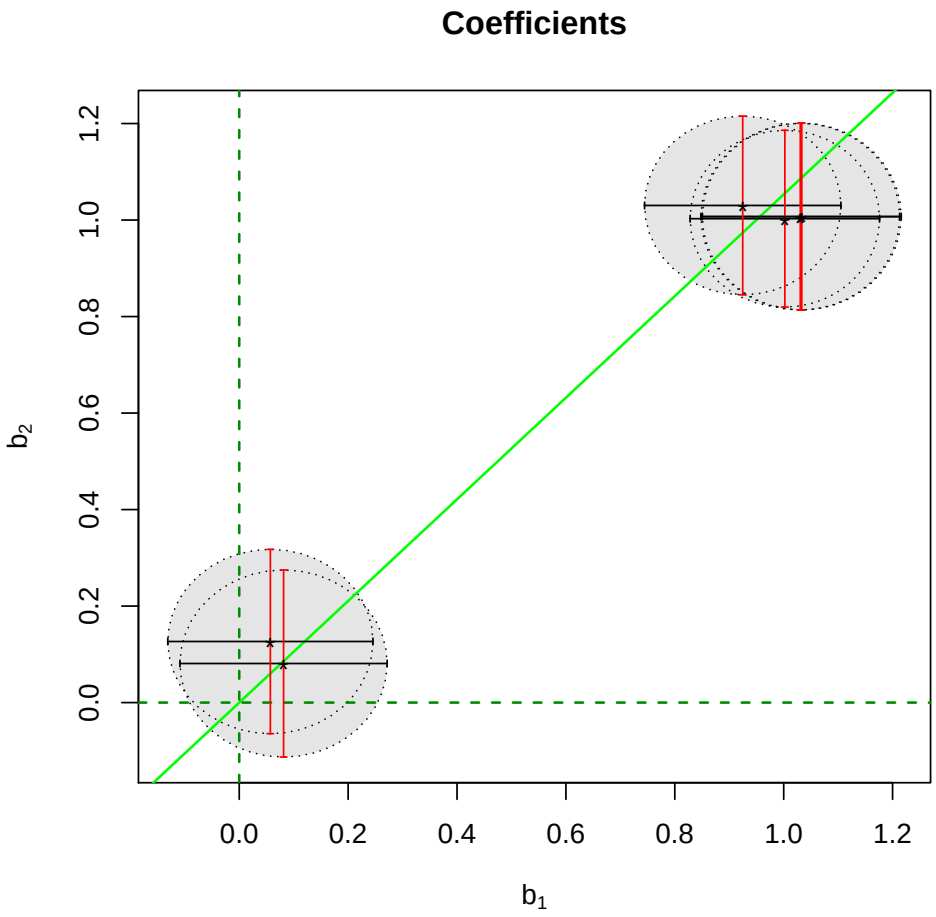
```
> ct.pcs.bayes <- coloc.test(pcs.1,pcs.2, bayes=TRUE)

.................

> ci(ct.pcs.bayes)

$eta.mode
[1] 1.052926

$lower
[1] 0.8827611

$upper
[1] 1.25793

$level.observed
[1] 0.9428057

$interior
[1] TRUE
```

3.2 Bayesian model averaging

This approach appears simpler. There is no need to do any preparatory work, you require only a single function:

```
> ct.bma <- coloc.bma(data@df1, data@df2,
+                     family1="gaussian", family2="gaussian",
+                     plot.coeff=TRUE)

Dropped 0 of 10 SNPs due to LD: r2 > 0.95
 10 SNPs remain.
Restricting model space.
 8 SNPs have single SNP posterior probabilities < 0.01
Models containing only these SNPs will not be explored.
Fitting 17 multi SNP models to dataset 1
Fitting 17 multi SNP models to dataset 2
Averaging coloc testing over 3 models with posterior probabilities >= 6.1e-192
...
```

```
> ct.bma.bayes <- coloc.bma(data@df1, data@df2,
+                           family1="gaussian", family2="gaussian",
+                           bayes=TRUE)

Dropped 0 of 10 SNPs due to LD: r2 > 0.95
 10 SNPs remain.
Restricting model space.
 8 SNPs have single SNP posterior probabilities < 0.01
Models containing only these SNPs will not be explored.
Fitting 17 multi SNP models to dataset 1
Fitting 17 multi SNP models to dataset 2
Averaging coloc testing over 3 models with posterior probabilities >= 6.1e-192
..................

> ct.bma

  eta.hat         n ppp.value
1.0530774 2.0000000 0.5683149

> ci(ct.bma.bayes)

$eta.mode
[1] 1.05821

$lower
[1] 0.8872175

$upper
[1] 1.264404

$level
[1] 0.95

$interior
[1] TRUE
```



The `family1/2` parameters are used to specify the model and link function, ="gaussian"= (with identity link) or ="binomial"= (with logit link) are accepted values. The plot here shows all models considered simultaneously, and does not attempt to discriminate between models with strong and weak support. It's rather confusing, and is off by default.

However, `coloc.bma()` is doing quite some work to cover the model space efficiently, and it is important to understand how it does this. First, the `r2.trim` parameter is used to "tag" the SNPs - a subset of SNPs are selected so that no pair have $r^2 >$ `r2.trim`. The default value is 0.95 and the idea is that models containing SNPs with very similar genotypes provide little additional information, so the p value need be averaged over only one of each such group. Lower values of `r2.trim` will produce a sparser model space and so decrease computation. Second, the `thr` parameter is used to discard SNPs which are uninformative with regards the phenotype, that is, if pp_{ij} is the posterior probability of inclusion in single SNP models for SNP i, trait j, the set of discarded SNPs is formed by those for which $pp_{i1} <$ `thr` and $pp_{i2} <$ `thr`. Models containing **only** SNPs from this set will be ignored. Note that models containing one SNP from this set and one SNP *not* in the set **will** be evaluated.

Finally, you should tell `coloc.bma()` how many SNPs should be included in each model. The default is `nsnps=2`, 3 appears slightly more powerful but will generally require considerably more

computation, whilst values of 4 and above are both unlikely to provide more information and very unlikely to be computed in any reasonable time for interactive work.

3.3 Using Bayes Factors to compare specific values of η

It may be that specific values of η are of interest. For example, when comparing eQTLs in two tissues, or when comparing risk of two related diseases, the value $\eta = 1$ is of particular interest. In proportional testing, we can use Bayes Factors to compare the support for different values of η. Eg

```
> ## compare individual values of eta
> ct.pcs <- coloc.test(pcs.1,pcs.2, bayes.factor=c(-1,0,1))

.................

> bf(ct.pcs)

          values.against
values.for            -1            0             1
        -1  1.000000e+00 5.425920e-52 3.847975e-103
        0   1.843006e+51 1.000000e+00  7.091840e-52
        1  2.598769e+102 1.410071e+51  1.000000e+00

> ## compare ranges of eta
> ct.bma <- coloc.bma(data@df1, data@df2,
+                     family1="gaussian", family2="gaussian",
+                     bayes.factor=list(c(-0.1,1), c(0.9,1.1)))

Dropped 0 of 10 SNPs due to LD: r2 > 0.95
 10 SNPs remain.
Restricting model space.
 8 SNPs have single SNP posterior probabilities < 0.01
Models containing only these SNPs will not be explored.
Fitting 17 multi SNP models to dataset 1
Fitting 17 multi SNP models to dataset 2
Averaging coloc testing over 3 models with posterior probabilities >= 6.1e-192
..................

> bf(ct.bma)

             values.against
values.for    c(-0.1, 1) c(0.9, 1.1)
  c(-0.1, 1)     1.00000  0.05150753
  c(0.9, 1.1)   19.41464  1.00000000
```

4 (Approximate) Bayes Factor colocalisation analyses

4.1 Introduction

The idea behind the ABF analysis is that the association of each trait with SNPs in a region may be summarised by a vector of 0s and at most a single 1, with the 1 indicating the causal SNP (so, assuming a single causal SNP for each trait). The posterior probability of each possible configuration can be calculated and so, crucially, can the posterior probabilities that the traits share their configurations. This allows us to estimate the support for the following cases:

- H_0: neither trait has a genetic association in the region
- H_1: only trait 1 has a genetic association in the region
- H_2: only trait 2 has a genetic association in the region
- H_3: both traits are associated, but with different causal variants
- H_4: both traits are associated and share a single causal variant

4.2 The basic `coloc.abf` function

The function `coloc.abf` is ideally suited to the case when only summary data are available, and requires, for each trait, either:

- p values for each SNP
- each SNP's minor allele frequency
- sample size
- ratio of cases:controls (if using a case-control trait)

or:

- regression coefficients for each SNP
- variance of these regression coefficients.

If regression coefficients and their variance are available, please use these, but we can also approximate Bayes Factors from p values and minor allele frequencies, although, note, such approximation can be less accurate when imputed data are used. NB, you can mix and match, depending on the data available from each study.

A wrapper function is avalailable to run all these steps but we will first generate the p values manually to give all the details. We use the snpStats library from Bioconductor to calculate the p values quickly.

```
> library(snpStats)
> Y1 <- data@df1$Y
> Y2 <- data@df2$Y
> X1 <- new("SnpMatrix",as.matrix(data@df1[,-1]))
```

```
coercing object of mode  numeric  to SnpMatrix

> X2 <- new("SnpMatrix",as.matrix(data@df2[,-1]))

coercing object of mode  numeric  to SnpMatrix

> p1 <- snpStats::p.value(single.snp.tests(phenotype=Y1, snp.data=X1),df=1)
> p2 <- snpStats::p.value(single.snp.tests(phenotype=Y2, snp.data=X2),df=1)
> maf <- col.summary(X2)[,"MAF"]
```

Note that we are using the second dataset in that case to compute the minor allele frequencies. This is unlikely to make any significant difference but one could have used dataset 1 instead. It is now possible to compute the probabilities of interest.

```
> my.res <- coloc.abf(dataset1=list(pvalues=p1,N=nrow(X1),type="quant"),
+                     dataset2=list(pvalues=p2,N=nrow(X2),type="quant"),
+                     MAF=maf)

PP.H0.abf PP.H1.abf PP.H2.abf PP.H3.abf PP.H4.abf
 5.02e-06  1.20e-04  4.31e-05  3.40e-05  1.00e+00
[1] "PP abf for shared variant: 100%"

> print(my.res[[1]])

       nsnps    PP.H0.abf    PP.H1.abf    PP.H2.abf    PP.H3.abf    PP.H4.abf
1.000000e+01 5.021713e-06 1.203555e-04 4.313383e-05 3.399227e-05 9.997975e-01
```

4.3 The wrapper function

Here, as we have simulated full genotype data, we can use the wrapper function `coloc.abf.datasets()` to combine all the steps shown above.

```
> ct.abf <- coloc.abf.datasets(data@df1, data@df2, response1="Y", response2="Y",
+                              type1="quant", type2="quant")

coercing object of mode  numeric  to SnpMatrix
coercing object of mode  numeric  to SnpMatrix
PP.H0.abf PP.H1.abf PP.H2.abf PP.H3.abf PP.H4.abf
 4.23e-06  1.20e-04  3.63e-05  3.37e-05  1.00e+00
[1] "PP abf for shared variant: 100%"
```

5 The difference between proportional and ABF approaches

So what are the differences between proportional and ABF approaches? Which should you choose?

Well, if you only have p values, then you must use ABF. But be aware that a single causal variant is assumed, and that for accurate inference, the causal variant needs to be included, so either very dense or imputed genotyping data is needed. The ABF approach has another big advantage

over proportional testing: being Bayesian, it allows you to evaluate support for each hypothesis, whereas with proportional testing, when the null of colocalisation is not rejected, you cannot be sure whether this reflects true colocalisation or a lack of power. The proportional approach is much less affected by spareness of genotyping data, with power only slightly decreased, but type 1 error rates unaffected.

The behaviour of the two approaches only really differs when there are two or more causal variants. When both are shared, proportional testing has the same type 1 error rate, and whilst ABF tends to still favour H_4 (shared variant), it tends to put some more weight on H_3 (distinct variants). But when at least one is specific to one trait and at least one is shared, their behaviour differs more substantially. Of course, neither approach was designed with this case in mind, and so there is no "right" answer, but it is instructive to understand their expected behaviour. Proportional testing tends to reject colocalisation, whilst ABF tends to favour sharing. Until the methods are extended to incorporate this specific case, it can be useful to compare the two approaches when complete and dense genotyping data are available. When the results differ, we have tended to identify a combination of shared and distinct causal variants. The ABF approach can still be applied in this case, if p values are available conditioning on the strongest signal, as demonstrated in our paper [4].

You can see more about the ABF approach on this blogpost.