colocRedRibbon: a RedRibbon based colocalization of GWAS and eQTL.

Theodora Papadoulou, Anthony Piron

2024-10-30

Abstract

Large genome-wide association studies (GWAS) and expression quantitative trait locus (eQTL) analyses can be integrated to better understand the genetic causality of complex traits and diseases. Colocalization methods allow to combine those datasets. One of the main challenges in colocalization analyses is the presence of multiple signals in GWAS regions around a gene. This multiplicity of signals can confound the colocalization process, making it difficult to pinpoint specific variants driving the associations. To address this, colocRedRibbon isolates the signal peaks of interest before conducting the colocalization analysis.

To demonstrate colocRedRibbon's basic workflow, we will perform a colocalization analysis using type 2 diabetes GWAS data and pancreatic islet eQTL data. We will use the real dataset "th", which is composed of a GWAS for type 2 diabetes (derived from the https://www.diagram-consortium.org/) and a human pancreatic eQTL colocalization study for the gene *TH*, encoding tyrosine hydroxylase (originating from http://tiger.bsc.es/). The region around the *TH* gene will be analyzed to identify shared genetic variants that influence both type 2 diabetes risk and *TH* gene expression in pancreatic islets.

Loading the package and the test dataset

We first load the package and the dataset with

```r
library(colocRedRibbon)
#> Loading required package: coloc
#> This is coloc version 5.2.3
#> Loading required package: data.table
#> Loading required package: ggplot2
#> Loading required package: ggpubr
#> Loading required package: RedRibbon
#> Loading required package: scales
#> Loading required package: ggrepel
## load data.table package for fast data.frame data structure
library(data.table)
## load the test dataset
data("th", package = "colocRedRibbon")
```

The `data()` function loads a `data.table` named `th.dt` into the current environment. Once the data is loaded, you can use the `head()` function to display the first few rows of the dataset.

Running the colocalization

Create an S3 colocRedRibbon object for the risk alleles decreasing expression,

rsid	pval.GWAS	n.GWAS	eaf.GWAS	or.GWAS	ea.GWAS	nea.GWAS	pval.eQTL	pos	n.eQTL	zscor
rs1003483	0.35	231420	0.510	1.0063199	T	G	0.68710	2167543	404	
rs1003484	0.64	231420	0.260	0.9965061	A	G	0.92180	2167618	404	
rs1003889	0.99	187126	0.011	0.9993002	T	G	0.58720	1970108	317	
rs1004446	0.38	231420	0.380	1.0059174	A	G	0.92870	2170143	404	
rs1005135	0.52	231420	0.210	0.9949130	C	G	0.08475	2539890	404	
rs1005236	0.16	231420	0.730	1.0110607	A	G	0.89080	2448519	404	

```
rrc.dec <- RedRibbonColoc(th.dt, risk="a", effect=`<=`,
                          columns=c(id="rsid", position="pos", a="pval.GWAS", b="pval.eQTL",
                                    a.n="n.GWAS", a.eaf="eaf.GWAS", a.dir="dir.GWAS",
                                    b.n="n.eQTL", b.eaf="eaf.eQTL", b.dir="zscore.eQTL"))
```

The risk allele is specified with the `risk` parameter, which can be either *NULL*, *'a'* or *'b'*. It represents the odds ratios for the analysis assigned to the *a* or *b* column. The `effect` parameter is set to `<=` to subselect the decreasing risk alleles. The `column` parameter defines the role of the `data.table` columns. The data.table should include the columns *id*, *a*, *b* and *position*, which represent the name of the SNP (Single Nucleotide Polymorphism), the p-value of the first analysis (here, type 2 diabetes GWAS), the p-value of the second analysis (here, human pancreatic islet eQTL), and the position of the SNP on the chromosome, respectively. In the following steps, RedRibbon will use the specified p-values for the overlap analysis. In addition to these essential columns, the data.table may include supplementary columns that facilitate the computation of co-localization.

Next, the co-localization can be computed with

```
## Run C. Wallace coloc()
rrc.dec <- coloc(rrc.dec)
```

Several optional parameters can be specified to customize the behavior of the coloc function. For more information see the documentation.

To extract the best SNP:

```
coloc.res["bestSnp"]
## Is this command correct?
## Anything else that we can extract?
```

The colocRedRibbon object

The RedRibbon overlap p-value is given by

```
rrc.dec$quadrants$whole$pvalue
#> [1] 2.621389e-41
```

and the overlapping variants are

```
rrc.dec$data[rrc.dec$quadrants$whole$positions]$id
#>  [1] "rs10743152" "rs10770140" "rs10770141" "rs10770142" "rs10770143"
#>  [6] "rs10840495" "rs10840496" "rs10840500" "rs11042965" "rs11042966"
#> [11] "rs11042976" "rs11042978" "rs11564711" "rs4929964"  "rs4929965"
#> [16] "rs4929966"  "rs4930046"  "rs7115640"  "rs7119275"  "rs7128097"
#> [21] "rs72853903" "rs7482891"
```

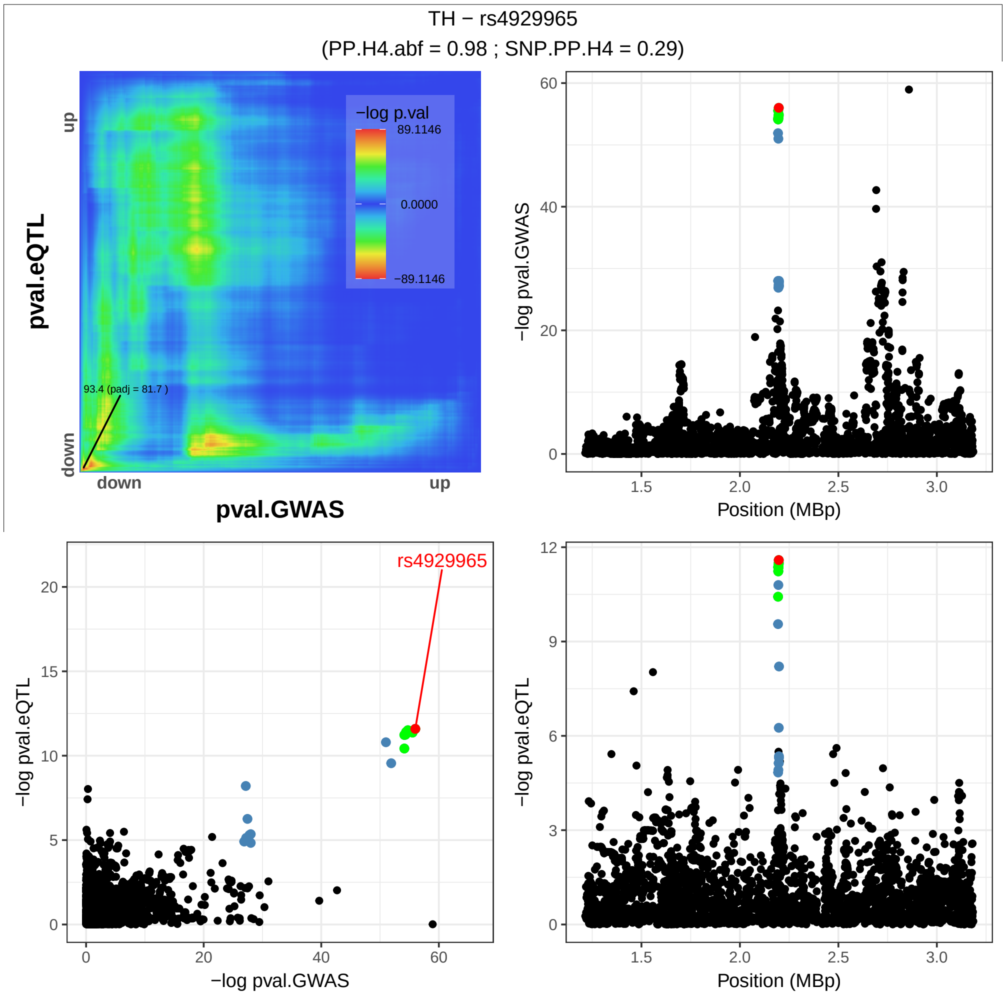
The coloc structure contains

```
rrc.dec$coloc
#> $bestSnp
#> [1] "rs4929965"
#> 
#> $PP.H4.abf
#> PP.H4.abf 
#> 0.9799012 
#> 
#> $SNP.PP.H4
#> [1] 0.2890225
#> 
#> $ncredibleSet99
#> [1] 12
#> 
#> $credibleSet99
#>  [1] "rs4929965"  "rs10770142" "rs4929964"  "rs7482891"  "rs10840496"
#>  [6] "rs11042966" "rs10743152" "rs10840495" "rs7128097"  "rs7115640" 
#> [11] "rs7119275"  "rs10770141"
#> 
#> $all.snps
#>             snp position pvalues.df1 MAF.df1  N.df1        V.df1     z.df1
#>  1:  rs4929965  2197286     4.8e-25    0.38 231420 9.170517e-06 10.336862
#>  2: rs10770142  2194420     7.7e-25    0.38 231420 9.170517e-06 10.291460
#>  3:  rs4929964  2197132     1.4e-24    0.38 231420 9.170517e-06 10.233744
#>  4:  rs7482891  2197112     1.7e-24    0.38 231420 9.170517e-06 10.214931
#>  5: rs10840496  2195844     1.9e-24    0.38 231420 9.170517e-06 10.204138
#>  6: rs11042966  2195538     1.9e-24    0.38 231420 9.170517e-06 10.204138
#>  7: rs10743152  2195981     1.9e-24    0.38 231420 9.170517e-06 10.204138
#>  8: rs10840495  2195837     2.3e-24    0.38 231420 9.170517e-06 10.185573
#>  9:  rs7128097  2195045     2.3e-24    0.38 231420 9.170517e-06 10.185573
#> 10:  rs7115640  2194914     2.6e-24    0.38 231420 9.170517e-06 10.173642
#> 11:  rs7119275  2194810     3.1e-24    0.38 231420 9.170517e-06 10.156502
#> 12: rs10770141  2193840     3.1e-24    0.39 231420 9.081857e-06 10.156502
#> 13: rs10770143  2195267     7.1e-23    0.39 231420 9.081857e-06  9.846461
#> 14: rs10770140  2193597     2.9e-23    0.39 231420 9.081857e-06  9.936085
#> 15: rs72853903  2198665     1.6e-12    0.31 231420 1.010086e-05  7.065534
#> 16:  rs4929966  2197436     1.2e-12    0.28 231420 1.071713e-05  7.105371
#> 17: rs11042976  2198259     6.9e-13    0.48 231420 8.656145e-06  7.181402
#> 18: rs11042978  2198418     8.6e-13    0.48 231420 8.656145e-06  7.151236
#> 19: rs11564711  2194062     6.9e-13    0.48 231420 8.656145e-06  7.181402
#> 20:  rs4930046  2197148     1.3e-12    0.48 231420 8.656145e-06  7.094308
#> 21: rs10840500  2196910     1.5e-12    0.48 231420 8.656145e-06  7.074489
#> 22: rs11042965  2195288     2.1e-12    0.48 231420 8.656145e-06  7.027677
#>             snp position pvalues.df1 MAF.df1  N.df1        V.df1     z.df1
#>         r.df1 lABF.df1 pvalues.df2 MAF.df2 N.df2       V.df2    z.df2     r.df2
#>  1: 0.9995926 49.50075   9.255e-06    0.38   404 0.005253072 4.433885 0.8107211
#>  2: 0.9995926 49.03266   1.155e-05    0.38   404 0.005253072 4.385914 0.8107211
#>  3: 0.9995926 48.44058   1.041e-05    0.38   404 0.005253072 4.408477 0.8107211
#>  4: 0.9995926 48.24831   1.004e-05    0.38   404 0.005253072 4.416310 0.8107211
#>  5: 0.9995926 48.13817   1.112e-05    0.38   404 0.005253072 4.394164 0.8107211
#>  6: 0.9995926 48.13817   1.132e-05    0.38   404 0.005253072 4.390289 0.8107211
#>  7: 0.9995926 48.13817   1.179e-05    0.38   404 0.005253072 4.381435 0.8107211
#>  8: 0.9995926 47.94898   1.113e-05    0.38   404 0.005253072 4.393968 0.8107211
#>  9: 0.9995926 47.94898   1.172e-05    0.38   404 0.005253072 4.382732 0.8107211
#> 10: 0.9995926 47.82757   1.313e-05    0.38   404 0.005253072 4.357927 0.8107211
#> 11: 0.9995926 47.65341   1.326e-05    0.38   404 0.005253072 4.355770 0.8107211
#> 12: 0.9995965 47.64876   2.967e-05    0.39   404 0.005202286 4.175985 0.8122073
#> 13: 0.9995965 44.54914   2.046e-05    0.39   404 0.005202286 4.259811 0.8122073
#> 14: 0.9995965 45.43528   7.092e-05    0.39   404 0.005202286 3.973178 0.8122073
#> 15: 0.9995513 21.09513   2.730e-04    0.31   404 0.005785992 3.639658 0.7954467
#> 16: 0.9995239 21.40618   1.919e-03    0.28   404 0.006139007 3.102490 0.7856418
#> 17: 0.9996154 21.84466   4.742e-03    0.48   404 0.004958429 2.824057 0.8194205
#> 18: 0.9996154 21.62856   4.996e-03    0.48   404 0.004958429 2.807292 0.8194205
#> 19: 0.9996154 21.84466   7.957e-03    0.48   404 0.004958429 2.653889 0.8194205
#> 20: 0.9996154 21.22323   5.884e-03    0.48   404 0.004958429 2.754177 0.8194205
#> 21: 0.9996154 21.08288   5.926e-03    0.48   404 0.004958429 2.751848 0.8194205
#> 22: 0.9996154 20.75293   7.348e-03    0.48   404 0.004958429 2.680647 0.8194205
#>         r.df1 lABF.df1 pvalues.df2 MAF.df2 N.df2       V.df2    z.df2     r.df2
#>     lABF.df2 internal.sum.lABF    SNP.PP.H4 SNP.PP.H4.cumsum
#>  1: 7.136852          56.63760 2.890225e-01        0.2890225
#>  2: 6.965345          55.99800 1.524606e-01        0.4414830
#>  3: 7.045782          55.48636 9.140217e-02        0.5328852
#>  4: 7.073802          55.32211 7.755740e-02        0.6104426
#>  5: 6.994708          55.13287 6.418574e-02        0.6746284
#>  6: 6.980911          55.11908 6.330628e-02        0.7379346
#>  7: 6.949430          55.08760 6.134434e-02        0.7992790
#>  8: 6.994012          54.94299 5.308502e-02        0.8523640
#>  9: 6.954038          54.90301 5.100483e-02        0.9033688
#> 10: 6.866150          54.69372 4.137307e-02        0.9447419
#> 11: 6.858530          54.51194 3.449607e-02        0.9792380
#> 12: 6.245771          53.89453 1.860508e-02        0.9978431
#> 13: 6.532944          51.08209 1.117385e-03        0.9989604
#> 14: 5.574603          51.00989 1.039554e-03        1.0000000
#> 15: 4.475223          25.57036 9.302493e-15        1.0000000
#> 16: 3.011022          24.41720 2.936230e-15        1.0000000
#> 17: 2.411768          24.25643 2.500162e-15        1.0000000
#> 18: 2.373088          24.00165 1.937845e-15        1.0000000
#> 19: 2.029849          23.87451 1.706486e-15        1.0000000
#> 20: 2.252061          23.47530 1.144792e-15        1.0000000
#> 21: 2.246808          23.32969 9.896725e-16        1.0000000
#> 22: 2.088332          22.84127 6.072547e-16        1.0000000
#>     lABF.df2 internal.sum.lABF    SNP.PP.H4 SNP.PP.H4.cumsum
```

Plotting the level map

Co-localization results can be visualized using the helper function *ggRedRibbonColoc*. This function helps generate plots that illustrate the colocalization analysis. By evaluating the p-values of the overlap map, users can easily interpret the relationships between SNPs associated with an increased risk for disease and those responsible for gene expression variation.

```
gg <- ggRedRibbonColoc(rrc.dec, shortid = "TH")
gg
```



For additional optional parameters to customize the visualization of colocalization results please see the function documentation in R (with `?ggRedRibbonColoc`).

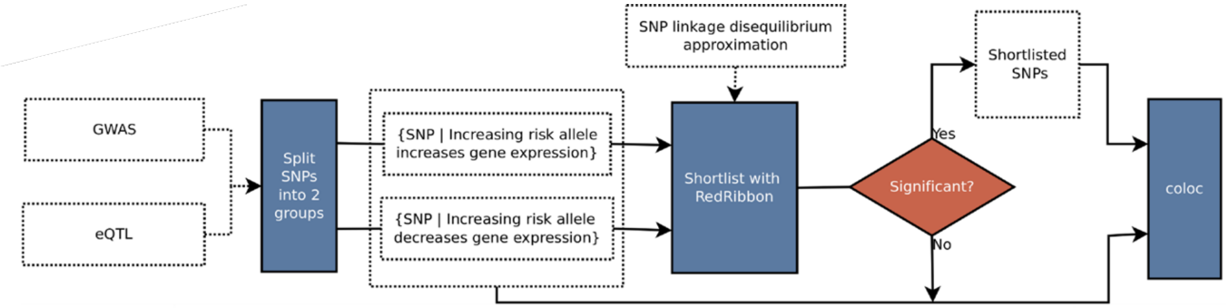
The gg variable contains a standard ggplot2 object, which can be modified and customized using the ggplot2 parameters in R.

```
class(gg)
#> [1] "gg"     "ggplot"
```

colocRedRibbon workflow

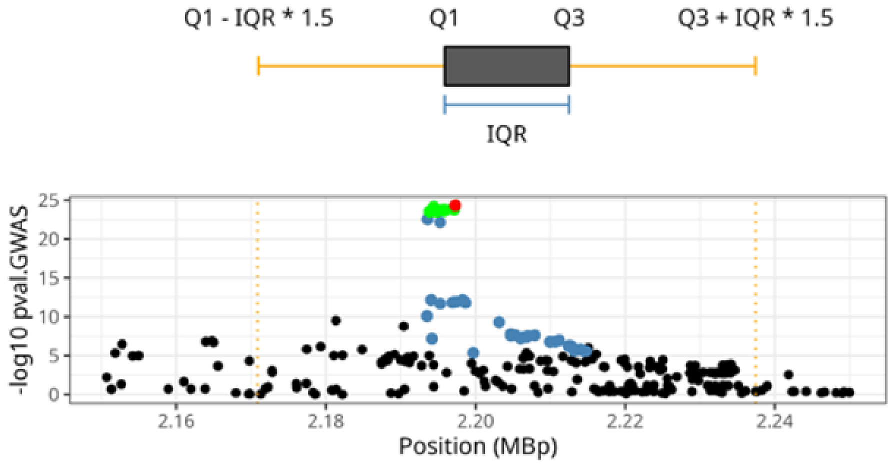
colocRedRibbon is a method designed to identify common causal candidates by examining the co-localization of GWAS and eQTL. This approach aims to pinpoint variants that are linked to both disease risk and variation in gene expression.

The method employs a two-step approach for shortlisting variants: - **Risk allele effect step:** In this step, variants are categorized into two distinct groups based on their direction of effect on gene expression, i.e., down- or upregulating. The upregulating variant set comprises the variants whose risk alleles increase gene expression, and the downregulating set risk alleles that decrease gene expression. Each of these variant sets is analyzed independently in subsequent steps. - **RedRibbon Overlap Step:** In this step, the RedRibbon rank-rank hypergeometric overlap method is applied on both GWAS and eQTL variants, which are ranked according to their P-values. This analysis examines the potential overlap between the two ranked lists. If a significant overlap is detected by RedRibbon, these shortlisted SNPs are further analyzed by the coloc package. If no significant overlap is found, the coloc method is still applied to the two effect sets without the preliminary overlap shortlisting.


GWAS
eQTL
Split SNPs into 2 groups
{SNP | Increasing risk allele increases gene expression}
{SNP | Increasing risk allele decreases gene expression}
SNP linkage disequilibrium approximation
Shortlist with RedRibbon
Significant?
Yes
No
Shortlisted SNPs
coloc

IQR mode

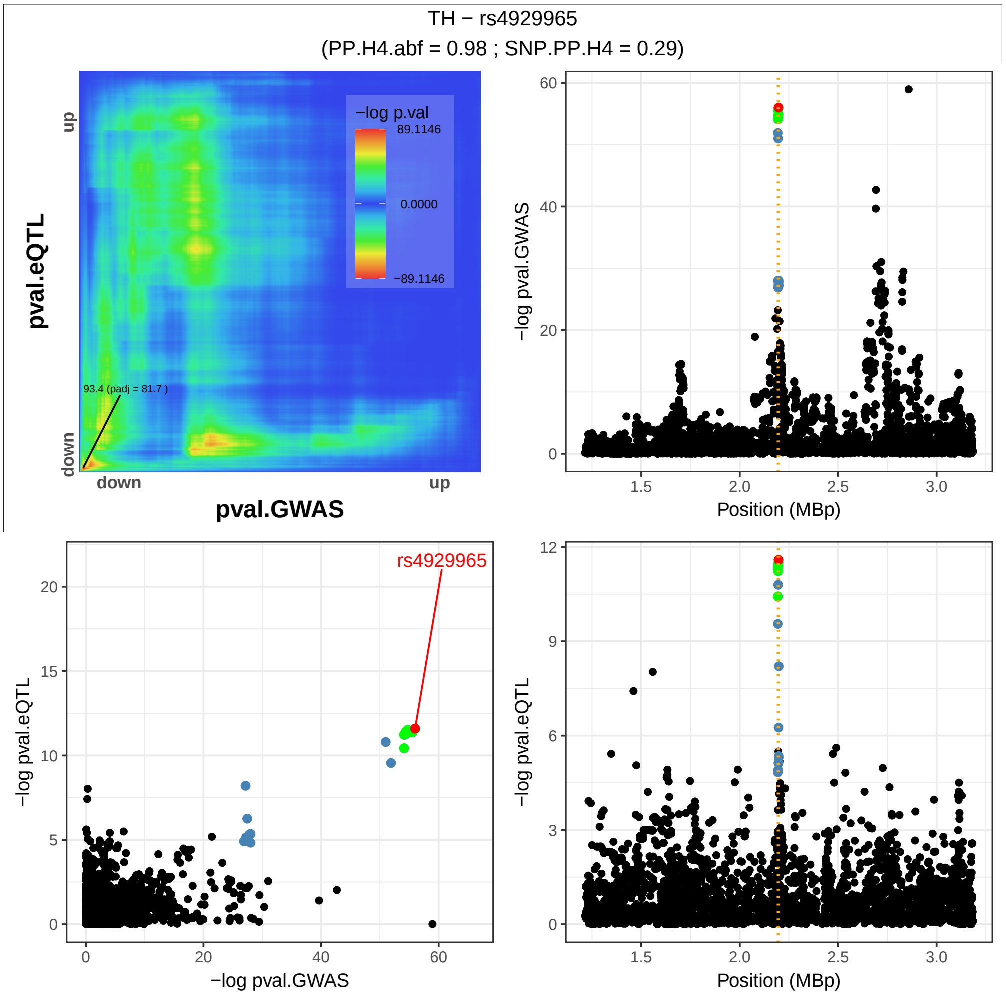
The area between the 25th percentile (first quartile) and the 75th percentile (third quartile) of the chromosomal positions of the core set is referred to as the interquartile range (IQR). IQR method includes the overlapping variants from *RedRibbon* and the variants in the IQR. The method delimits the region scrutiny.



The IQR mode is activated by setting the `region.mode` parameter,

```
## Run C. Wallace coloc()
rrcIQR.dec <- coloc(rrc.dec, region.mode = "IQR")
```

```
gg <- ggRedRibbonColoc(rrcIQR.dec, shortid = "TH")
gg
```



sessionInfo()

```
sessionInfo()
#> R version 4.2.2 Patched (2022-11-10 r83330)
#> Platform: x86_64-pc-linux-gnu (64-bit)
#> Running under: Debian GNU/Linux 12 (bookworm)
#>
#> Matrix products: default
#> BLAS:   /usr/lib/x86_64-linux-gnu/blas/libblas.so.3.11.0
#> LAPACK: /usr/lib/x86_64-linux-gnu/lapack/liblapack.so.3.11.0
#>
#> locale:
#>  [1] LC_CTYPE=en_US.UTF-8       LC_NUMERIC=C
#>  [3] LC_TIME=en_US.UTF-8        LC_COLLATE=en_US.UTF-8
#>  [5] LC_MONETARY=en_US.UTF-8    LC_MESSAGES=en_US.UTF-8
#>  [7] LC_PAPER=en_US.UTF-8       LC_NAME=C
#>  [9] LC_ADDRESS=C               LC_TELEPHONE=C
#> [11] LC_MEASUREMENT=en_US.UTF-8 LC_IDENTIFICATION=C
#>
#> attached base packages:
```

```
#> [1] stats     graphics  grDevices utils     datasets  methods   base
#> 
#> other attached packages:
#> [1] colocRedRibbon_0.2-1 RedRibbon_1.1-1      ggrepel_0.9.3
#> [4] scales_1.2.1         ggpubr_0.6.0         ggplot2_3.4.1
#> [7] data.table_1.14.8    coloc_5.2.3          kableExtra_1.4.0
#> 
#> loaded via a namespace (and not attached):
#>  [1] Rcpp_1.0.10        svglite_2.1.1      lattice_0.20-45    tidyr_1.3.0
#>  [5] assertthat_0.2.1   digest_0.6.31      utf8_1.2.3         R6_2.5.1
#>  [9] plyr_1.8.8         backports_1.4.1    evaluate_0.20      pillar_1.8.1
#> [13] rlang_1.0.6        rstudioapi_0.14    irlba_2.3.5.1      car_3.1-1
#> [17] Matrix_1.5-3       rmarkdown_2.20     labeling_0.4.2     stringr_1.5.0
#> [21] munsell_0.5.0      mixsqp_0.3-48      broom_1.0.3        compiler_4.2.2
#> [25] xfun_0.37          pkgconfig_2.0.3    systemfonts_1.0.4  htmltools_0.5.4
#> [29] tidyselect_1.2.0   tibble_3.1.8       gridExtra_2.3      matrixStats_0.63.0
#> [33] reshape_0.8.9      fansi_1.0.4        viridisLite_0.4.1  crayon_1.5.2
#> [37] dplyr_1.0.10       withr_2.5.0        grid_4.2.2         gtable_0.3.1
#> [41] lifecycle_1.0.3    DBI_1.1.3          magrittr_2.0.3     cli_3.6.0
#> [45] stringi_1.7.12     carData_3.0-5      farver_2.1.1       ggsignif_0.6.4
#> [49] viridis_0.6.2      xml2_1.3.3         generics_0.1.3     vctrs_0.5.2
#> [53] cowplot_1.1.1      tools_4.2.2        glue_1.6.2         susieR_0.12.35
#> [57] purrr_1.0.1        abind_1.4-5        fastmap_1.1.1      yaml_2.3.7
#> [61] colorspace_2.1-0   rstatix_0.7.2      knitr_1.42
```