Introduction to coMethDMR

Lissette Gomez, Gabriel Odom, Lanyu Zhang, Lily Wang

September 16, 2019

coMethDMR is an R package that identifies genomic regions that are both co-methylated and differentially methylated in Illumina array datasets. Instead of testing all CpGs within a genomic region, coMethDMR carries out an additional step that selects co-methylated sub-regions first without using any outcome information. Next, coMethDMR tests association between methylation within the sub-region and continuous phenotype using a random coefficient mixed effects model, which models both variations between CpG sites within the region and differential methylation simultaneously.

1. Quick start

1.1 Installation

The latest version can be installed by

```
library(devtools)
install_github ("lissettegomez/coMethDMR")
```

After installation, the coMethDMR package can be loaded into R using:

```
library(coMethDMR)
```

1.2 Datasets

The input of coMethDMR are methylation beta values. We assume quality control and normalization of the methylation dataset have been performed, by R packages such as minfi or RnBeads. For illustration, we use a subset of prefrontal cortex methylation data (GEO GSE59685) from a recent Alzheimer's disease epigenome-wide association study which was described in Lunnon et al. (2014). This example dataset contains beta values for 8552 CpGs on chromosome 22 for a random selection of 20 subjects.

```
data(betasChr22_df)
betasChr22_df [1:5, 1:5]
```

```
##            GSM1443279 GSM1443663 GSM1443434 GSM1443547 GSM1443577
## cg00004192  0.9249942  0.8463296  0.8700718  0.9058205  0.9090382
## cg00004775  0.6523025  0.6247554  0.7573476  0.6590817  0.6726261
## cg00012194  0.8676339  0.8679048  0.8484754  0.8754985  0.8484458
## cg00013618  0.9466056  0.9475467  0.9566493  0.9588431  0.9419563
## cg00014104  0.3932388  0.5525716  0.4075900  0.3997278  0.3216956
```

The corresponding phenotype dataset included variables stage (Braak AD stage), subject.id, slide (batch effect), Sex, Sample and age.brain (age of the brain donor). Please note the phenotype file needs to have a variable called "Sample" that will be used by coMethDMR to link to the methylation dataset.

```
data(pheno_df)
head(pheno_df)
```

```
##    stage subject.id         sex     Sample age.brain       slide
## 3      0          1 Sex: FEMALE GSM1443251        82  6042316048
```

```
## 8      2          2 Sex: FEMALE GSM1443256        82  6042316066
## 10    NA          3   Sex: MALE GSM1443258        89  6042316066
## 15     1          4 Sex: FEMALE GSM1443263        81  7786923107
## 21     2          5 Sex: FEMALE GSM1443269        92  6042316121
## 22     1          6   Sex: MALE GSM1443270        78  6042316099
```

1.3 A quick work through of coMethDMR

For illustration, suppose we are interested in identifying co-methylated genomic regions associated with AD stages (`stage` treated as a linear variable). Here we demonstrate analysis of genomic regions mapped to CpG islands on chromosome 22. However the workflow can be similarly conducted for other types of genomic regions. See details in Section 2.1 below for gene based pipeline that tests genic and intergenic regions.

There are several steps: (1) obtain CpGs located closely (see details in Section 2.1 below) in genomic regions mapped to CpG islands, (2) identify co-methylated regions, and (3) test co-methylated regions against the outcome variable AD stage.

For the first step, we use the following commands:

```
CpGisland_ls <- readRDS(
  system.file (
    "extdata",
    "CpGislandsChr22_ex.RDS",
    package = 'coMethDMR',
    mustWork = TRUE
    )
)
```

Here, `CpGisland_ls` is a list of 20 items, with each item of the list including a group of CpG probe IDs located closely within a particular CpG island region. Section 2.1 discusses how to import additional types of genomic regions.

Next, we identify co-methylated regions based on Mvalues.

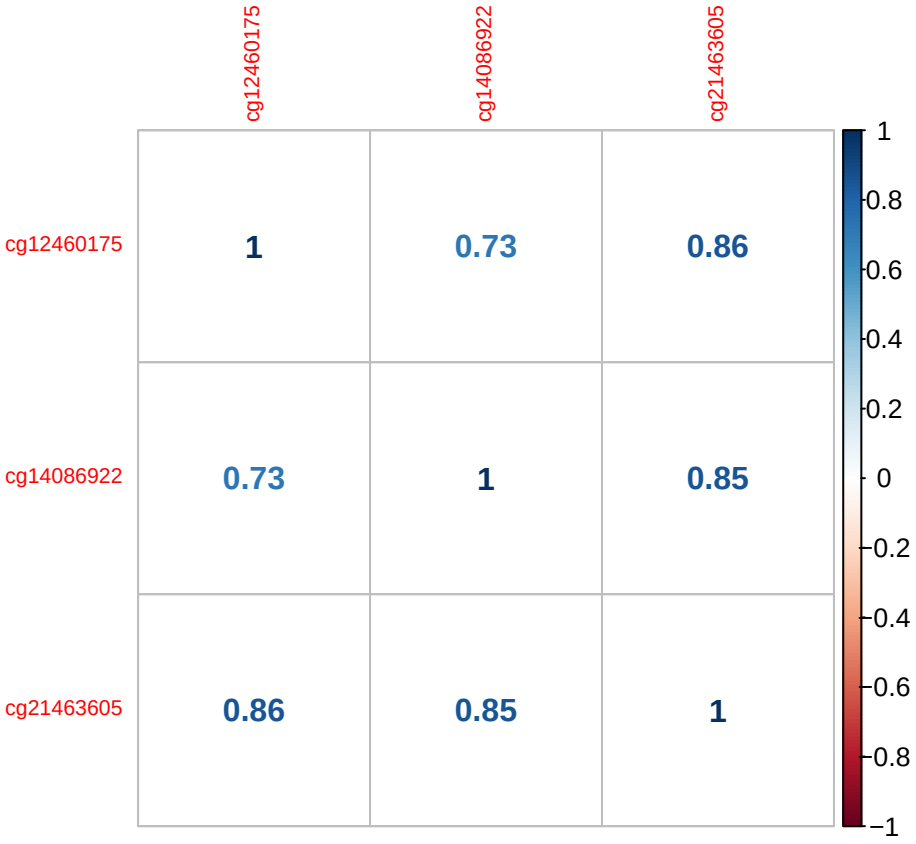
```
coMeth_ls <- CoMethAllRegions (
  dnam = betasChr22_df,
  betaToM = TRUE, #converts to mvalues
  CpGs_ls = CpGisland_ls,
  arrayType = "450k",
  returnAllCpGs = FALSE,
  output = "CpGs"
)

coMeth_ls
```

```
## $`chr22:18268062-18268249`
## [1] "cg12460175" "cg14086922" "cg21463605"
##
## $`chr22:18324579-18324769`
## [1] "cg19606103" "cg14031491" "cg03816851"
##
## $`chr22:18531243-18531447`
## [1] "cg25257671" "cg06961233" "cg08819022"
```

`coMeth_ls` is list with that contains groups of CpG probeIDs corresponding to co-methylated regions. Three comethylated regions were identified in this example.

If we want to look at co-methylation within the first co-methylated region:

```
WriteCorrPlot <- function (beta_mat){

  require (corrplot)
  require (coMethDMR)

  CpGs_char <- row.names (beta_mat)

  CpGsOrd_df <- OrderCpGsByLocation(
    CpGs_char, arrayType=c("450k"), output = "dataframe"
  )

  betaOrdered_mat <- t(beta_mat [CpGsOrd_df$cpg ,])

  corr <- cor (
    betaOrdered_mat, method = "spearman", use = "pairwise.complete.obs"
  )

  corrplot(corr, method="number", number.cex = 1, tl.cex = 0.7)
}

# subsetting beta values to include only co-methylated probes
betas_df <- subset(
  betasChr22_df,
  row.names(betasChr22_df) %in% coMeth_ls [[1]]
)

WriteCorrPlot (betas_df)
```



Next, we test these co-methylated regions against `stage` using a random coefficient model (more details in section 2.3 below).

Some messages are generated during mixed models fitting, which are saved to the file specified by `outLogFile`. The interpretations of these messages can be found in the FAQs at the end of this document (see Section 3, item (1) and (2)).

```
out_df <- lmmTestAllRegions(
   betas = betasChr22_df,
   region_ls = coMeth_ls,
   pheno_df,
   contPheno_char = "stage",
   covariates_char = NULL,
   modelType = "randCoef",
   arrayType = "450k",

   # generates a log file in the current directory
   outLogFile = paste0("lmmLog_", Sys.Date(), ".txt")
)
```

```
## messages for mixed model fittings are in file lmmLog_2019-09-21.txt
```

```
## Warning in checkConv(attr(opt, "derivs"), opt$par, ctrl =
## control$checkConv, : Model failed to converge with max|grad| = 0.0033928
## (tol = 0.002, component 1)
```

```
out_df
```

```
##    chrom     start       end nCpGs     Estimate      StdErr      Stat
```

```
## 1 chr22 18268062 18268249     3 -0.06678558 0.03883719 -1.719630
## 2 chr22 18324579 18324769     3  0.03549924 0.02884538  1.230673
## 3 chr22 18531243 18531447     3 -0.05181161 0.05082667 -1.019378
##       pValue       FDR
## 1 0.08549977 0.2564993
## 2 1.00000000 1.0000000
## 3 0.30802335 0.4620350
```

Here `out_df` is a data frame of genomic regions, with corresponding p-values and false discovery rate (FDRs) from the random coefficient mixed model.

We can annotate these results by adding corresponding genes and probes mapped to the genomic regions.

```
outAnno_df <- AnnotateResults(
  lmmRes_df = out_df,
  arrayType = "450k"
)

outAnno_df
```

```
##   chrom    start      end nCpGs    Estimate     StdErr      Stat
## 1 chr22 18268062 18268249     3 -0.06678558 0.03883719 -1.719630
## 2 chr22 18324579 18324769     3  0.03549924 0.02884538  1.230673
## 3 chr22 18531243 18531447     3 -0.05181161 0.05082667 -1.019378
##       pValue       FDR UCSC_RefGene_Group UCSC_RefGene_Accession
## 1 0.08549977 0.2564993
## 2 1.00000000 1.0000000               Body NM_001122731;NM_015241
## 3 0.30802335 0.4620350
##   UCSC_RefGene_Name Relation_to_Island
## 1                              Island
## 2            MICAL3            Island
## 3                              Island
```

To further examine the significant regions, we can also extract individual CpG p-values within these significant regions. For example, for the most significant region `chr22:18268062-18268249`,

```
outCpGs_df <- CpGsInfoOneRegion(
 regionName_char = "chr22:18268062-18268249",
 betas_df = betasChr22_df,
 pheno_df, contPheno_char = "stage",
 covariates_char = NULL,
 arrayType = "450k"
)

outCpGs_df
```

```
##                   Region        cpg   chr      pos slopeEstimate
## 1 chr22:18268062-18268249 cg12460175 chr22 18268062       -0.0329
## 2 chr22:18268062-18268249 cg14086922 chr22 18268239       -0.0724
## 3 chr22:18268062-18268249 cg21463605 chr22 18268249       -0.0951
##   slopePval UCSC_RefGene_Name UCSC_RefGene_Accession UCSC_RefGene_Group
## 1    0.3785
## 2    0.0667
## 3    0.0255
```

These CpGs mapped to intergenic regions, so there are no gene names associated with the probes. For genic regions such as `chr22:19709548-19709755`, we would have results such as the following:

```
CpGsInfoOneRegion(
  regionName_char = "chr22:19709548-19709755",
  betas_df = betasChr22_df,
  pheno_df, contPheno_char = "stage",
  covariates_char = NULL,
  arrayType = "450k"
)
```

```
##                   Region        cpg   chr      pos slopeEstimate
## 1 chr22:19709548-19709755 cg04533276 chr22 19709548       -0.0656
## 2 chr22:19709548-19709755 cg20193802 chr22 19709696       -0.0346
## 3 chr22:19709548-19709755 cg05726109 chr22 19709755        0.0021
##   slopePval UCSC_RefGene_Name UCSC_RefGene_Accession UCSC_RefGene_Group
## 1    0.1529             SEPT5              NM_002688               Body
## 2    0.4808       SEPT5;GP1BB    NM_002688;NM_000407       Body;TSS1500
## 3    0.9585       SEPT5;GP1BB    NM_002688;NM_000407       Body;TSS1500
```

2. Details of coMethDMR workflow

2.1 Genomic regions tested in gene based pipeline

Genomic regions on the Illumina arrays can be defined based on their relations to genes or CpG Islands. To reduce redundancy in the tested genomic regions, we recommend first testing genic and intergenic regions, then add annotations to each genomic region for their relation to CpG islands.

In coMethDMR package, for 450k arrays, the relevant genomic regions to be analyzed are in files 450k_Gene_3_200.RDS and 450k_InterGene_3_200.RDS. For EPIC arrays, the relevant genomic regions are in files EPIC_Gene_3_200.RDS and EPIC_InterGene_3_200.RDS.

These files were created using the function WriteCloseByAllRegions, briefly, for genic regions, within each gene, we identified clusters of CpGs located closely (i.e. the maximum separation between any two consecutive probes is 200bp; maxGap = 200), and we required each cluster to have at least 3 CpGs (minCpGs = 3). For intergenic regions, we identified clusters CpGs similarly for each chromosome. To extract clusters of close-by CpGs from pre-defined genomic regions with different values of maxGap and minCpGs, the WriteCloseByAllRegions function can be used.

The pre-computed genomic regions can be accessed using the following commands. For geneic regions in 450k arrays,

```
gene_ls <- readRDS(
  system.file (
    "extdata",
    "450k_Gene_3_200.RDS",
    package = 'coMethDMR',
    mustWork = TRUE
    )
)
```

Here gene_ls is a list, with each item containing a character vector of CpGs IDs for a particular region in a gene.

Vignette # 2 illustrates how to leverage parallel computing via BiocParallel R package to make gene-based analysis fast.

2.2 When there are co-variate variables in dataset to consider

Before identifying co-methylated clusters, we recommend removing uninteresting technical and biological effects, so that the resulting co-methylated clusters are only driven by the biological factors we are interested in. This can be accomplished using the `GetResiduals` function.

For example, the following script computes residuals from linear model `Mvalues ~ age.brain + sex + slide`

```
resid_df <- GetResiduals(
  dnam = betasChr22_df,
  betaToM = TRUE, #converts to Mvalues for fitting linear model
  pheno_df = pheno_df,
  covariates_char = c("age.brain", "sex", "slide")
)
```

2.2 Algorithm for identifying co-methylated regions

Within each genomic region, coMethDMR identifies contiguous and co-methylated CpGs sub-regions without using any outcome information. To select these co-methylated sub-regions, we use the `rdrop` statistic, which is the correlation between each CpG with the sum of methylation levels in all other CpGs. The default is `rDropThresh_num = 0.4`. We recommend this setting based on our simulation study. Note that higher `rDropThresh_num` values lead to fewer co-methylated regions.

Again, for illustration, we use CpG islands. For example, if we are interested in identifying co-methylated sub-region within the first genomic region in `Cgi_ls`:

```
Cgi_ls <- readRDS(
  system.file (
    "extdata",
    "CpGislandsChr22_ex.RDS",
    package = 'coMethDMR',
    mustWork = TRUE
    )
)

coMeth_ls <- CoMethAllRegions (
  dnam = resid_df,
  betaToM = FALSE,
  CpGs_ls = Cgi_ls[1],
  arrayType = "450k",
  returnAllCpGs = FALSE,
  output = "CpGs"
)

coMeth_ls
```

```
## NULL
```

The results indicate there is no co-methylated sub-region within the first genomic region.

Next we look at a region (13th region in `Cgi_ls`) where there is a co-methylated sub-region:

```
coMeth_ls <- CoMethAllRegions (
  dnam = resid_df,
  betaToM = FALSE,
  CpGs_ls = Cgi_ls[13],
```

```
  arrayType = "450k",
  returnAllCpGs = FALSE,
  output = "CpGs"
)

coMeth_ls
```

```
## $`chr22:18268062-18268249`
## [1] "cg12460175" "cg14086922" "cg21463605"
```

`coMeth_ls` is a list, where each item is a list of CpG probe IDs for a co-methylated sub-region.

If we want to see the detailed output of the coMethDMR algorithm, that is, how the co-methylated region was obtained, we can specify `output = "dataframe"`:

```
coMethData_df <- CoMethAllRegions (
  dnam = resid_df,
  betaToM = FALSE,
  CpGs_ls = Cgi_ls[13],
  arrayType = "450k",
  returnAllCpGs = FALSE,
  output = "dataframe"
) [[1]]

coMethData_df
```

```
##                    Region        CpG   Chr  MAPINFO    r_drop keep
## 1 chr22:18267969-18268249 cg18370151 chr22 18267969 0.3137027    0
## 2 chr22:18267969-18268249 cg12460175 chr22 18268062 0.8336860    1
## 3 chr22:18267969-18268249 cg14086922 chr22 18268239 0.7973932    1
## 4 chr22:18267969-18268249 cg21463605 chr22 18268249 0.8775942    1
##   keep_contiguous
## 1               0
## 2               1
## 3               1
## 4               1
```

`coMethData_df` provides the details on how the co-methylated region was obtained: Here `keep = 1` if `rDropThresh_num > 0.4` (i.e. a co-methylated CpG), and `keep_contigous` indicates if the probe is in a contiguous co-methylated region. Note that only the last 3 CpGs constitutes the co-methylated cluster.

2.3 Models for testing genomic regions against a continuous phenotype

To test association between a continuous phenotype and methylation values in a contiguous co-methylated region, two mixed models have been implemented in the function `lmmTestAllRegions`: a random coefficient mixed model (`modelType = "randCoef"`) and a simple linear mixed model (`modelType = "simple"`).

The random coefficient mixed model includes both a systematic component that models the mean for each group of CpGs, and a random component that models how each CpG varies with respect to the group mean (random probe effects). It also includes random sample effects that model correlations between multiple probes within the same sample.

More specifically, the random coefficient model is `methylation M value ~ contPheno_char + covariates_char + (1|Sample) + (contPheno_char|CpG)`. The last term `(contPheno_char|CpG)` specifies both random intercepts and slopes for each CpG.

The simple linear mixed model includes all the terms in the random coefficient model except random probe effects.

The simple linear mixed model is

```
methylation M value ~ contPheno_char + covariates_char + (1|Sample)
```

To test one genomic region against the continuous phenotype `stage`, adjusting for `age.brain`:

```
lmmTestAllRegions(
  betas = betasChr22_df,
  region_ls = coMeth_ls[1],
  pheno_df,
  contPheno_char = "stage",
  covariates_char = "age.brain",
  modelType = "randCoef",
  arrayType = "450k"
)
```

```
##   chrom    start      end nCpGs    Estimate     StdErr      Stat
## 1 chr22 18268062 18268249     3 -0.07320788 0.03943577 -1.856383
##       pValue        FDR
## 1 0.06339902 0.06339902
```

If we don't want to adjust for any covariate effect, we can set `covariates_char` to `NULL`:

```
lmmTestAllRegions(
  betas = betasChr22_df,
  region_ls = coMeth_ls[1],
  pheno_df,
  contPheno_char = "stage",
  covariates_char = NULL,
  modelType = "randCoef",
  arrayType = "450k"
)
```

```
##   chrom    start      end nCpGs    Estimate     StdErr     Stat     pValue
## 1 chr22 18268062 18268249     3 -0.06678558 0.03883719 -1.71963 0.08549977
##          FDR
## 1 0.08549977
```

3. Frequently Asked Questions

(1) What happens when mixed model fails to coverge (i.e. the warning "Model failed to converge with…" is resulted for a particular genomic region)?

- In this case, the p-value for mixed model is set to 1. In our experiences with methylation datasets, genomic regions with strong signals typically converge. Convergence issues typically occurs when the amount of noise in data is high.

(2) When fitting mixed models with `lmmTestAllRegions` function, What does the message "boundary (singular) fit" mean?

- When mixed model is singular, at least one of the estimated variance components for intercepts or slopes random effects is 0, because there isn't enough variabilities in data to estimate the random effects. In this case, mixed model reduces to a fixed effects model. However, as our simulation studies have shown, the p-values obtained for these regions are still valid.

4. Reference

Lunnon K, Smith R, Hannon E, De Jager PL, Srivastava G, Volta M, Troakes C, Al-Sarraj S, Burrage J, Macdonald R, et al (2014) Methylomic profiling implicates cortical deregulation of ANK1 in Alzheimer's disease. Nat Neurosci 17:1164-1170.