jDMRgrid: a heuristic DMR caller for WGBS data using grid approach

Robert S. Piecyk, Rashmi R. Hazarika, Yadi Shahryary, Frank Johannes

2024-10-31

Contents

R Markdown

This is an R Markdown document. Markdown is a simple formatting syntax for authoring HTML, PDF, and MS Word documents. For more details on using R Markdown see http://rmarkdown.rstudio.com.

When you click the **Knit** button a document will be generated that includes both content as well as the output of any embedded R code chunks within the document.

```
setwd("/jlab/home/ming/rmarkdown")
```

1 Input files

For generation of region-level calls, jDMRgrid requires the following inputs.

1.1 Methimpute files:

Base-level methylome outputs (generated using the R package "Methimpute")

1.2 A metadata file containing description about samples

For population data-sets without replicates, listfiles.fn should have the structure below.

```
load(system.file("data", "listFiles1.RData", package = "jDMRgrid"))
listFiles1$file <- system.file("extdata", listFiles1$file, package = "jDMRgrid")
listFiles1
```

```
##                                                                                file
## 1 /home/ming/R/x86_64-pc-linux-gnu-library/4.4/jDMRgrid/extdata/toyData/methimpute_p1.txt
## 2 /home/ming/R/x86_64-pc-linux-gnu-library/4.4/jDMRgrid/extdata/toyData/methimpute_p2.txt
## 3 /home/ming/R/x86_64-pc-linux-gnu-library/4.4/jDMRgrid/extdata/toyData/methimpute_p3.txt
## 4 /home/ming/R/x86_64-pc-linux-gnu-library/4.4/jDMRgrid/extdata/toyData/methimpute_p4.txt
##       sample replicate
## 1 methylomeA      rep1
## 2 methylomeB      rep1
## 3 methylomeC      rep1
## 4 methylomeD      rep1
```

For pairwise control-treatment data-sets with replicates,additional columns "replicate" and "group" should be provided. See structure below.

```
load(system.file("data", "listFiles2.RData", package = "jDMRgrid"))
listFiles2$file <- system.file("extdata", listFiles2$file, package = "jDMRgrid")
listFiles2
```

```
##                                                                                file
## 1 /home/ming/R/x86_64-pc-linux-gnu-library/4.4/jDMRgrid/extdata/toyData/methimpute_p1.txt
## 2 /home/ming/R/x86_64-pc-linux-gnu-library/4.4/jDMRgrid/extdata/toyData/methimpute_p2.txt
## 3 /home/ming/R/x86_64-pc-linux-gnu-library/4.4/jDMRgrid/extdata/toyData/methimpute_p3.txt
## 4 /home/ming/R/x86_64-pc-linux-gnu-library/4.4/jDMRgrid/extdata/toyData/methimpute_p4.txt
##    sample replicate      group
## 1      WT      rep1    control
## 2      WT      rep2    control
## 3 mutant1      rep1 treatment1
## 4 mutant1      rep2 treatment1
```

2 Generate cytosine region calls from genome

2.1 Run jDMRgrid on a binned genome

This function uses a grid approach to bin the genome into equal sized bins. User specifies the window and step size as numeric values.

out.dir: PATH to output directory.

window: NUMERIC VALUE specifying bin size.

step: NUMERIC VALUE specifying step size. If bin and step size are equal, we are utilizing non-sliding window approach.

samplelist: DATAFRAME OBJECT containing information about file, sample and replicate. For control/treatment data an additional column specifying the replicates is required.

contexts: VECTOR or CHARACTER presenting sequence contexts of the cytosine. By default this option is set to c("CG", "CHG", "CHH", "CHH"). If you want to run for a single context such as CG, set it as "CG".

min.C: NUMERIC VALUE specifying percentile threshold based on empirical distribution of the cytosines across bins.

mincov: NUMERIC VALUE specifying minimum read coverage over cytosines. By default this option is set as 0.

include.intermediate: LOGICAL specifying whether or not the intermediate component should be included in the HMM model. By default this option is set as FALSE.

runName: CHARACTER as the name of the operation. By default this option is set to 'GridGenome'.

parallelApply: LOGICAL specifying if future.apply package should be used to use parallel operation. By default this option is set to FALSE.

numCores: NUMERIC VALUE specifying number of cores to perform parallel operation using foreach loop. By default this option is set to NULL.

if.Bismark: Logical if Bismark inputs (CX reports in txt format) are used. Default as FALSE. (logical)

FASTA.file: Path to the FASTA file; required if Bismark outputs are used. Please use the same FASTA as in Bismark analysis. Default as NULL. (char)

```
library(jDMRgrid)
runjDMRgrid(out.dir = "./population",
            window = 200, step = 50, samplelist = listFiles1,
            contexts = c("CG", "CHG", "CHH"), min.C = 10, mincov = 0,
            include.intermediate = TRUE,
            runName = "Arabidopsis")

runjDMRgrid(out.dir = "./replication",
            window = 200, step = 50, samplelist = listFiles2,
            contexts = c("CG", "CHG", "CHH"), min.C = 10, mincov = 0,
            include.intermediate = TRUE,
            runName = "Arabidopsis")
```

2.2 Output files of jDMR Grid approach

Region files containing state calls, methylation levels and posteriorMax will be generated for each sample and for each context.

```r
library(data.table)
jDMRgrid.out <- fread("./population/methimpute_p1_CG.txt", header = T)
head(jDMRgrid.out)
```

```
##    seqnames start   end context posteriorMax status rc.meth.lvl
##       <int> <int> <int>  <char>        <num> <char>       <num>
## 1:        1     1   200      CG      0.97962      U     0.04022
## 2:        1    51   250      CG      0.79295      U     0.12897
## 3:        1   101   300      CG      0.79295      U     0.12897
## 4:        1   151   350      CG      0.64649      M     0.80224
## 5:        1   201   400      CG      0.64649      M     0.80224
## 6:        1   251   450      CG      0.64649      M     0.80224
```

seqnames, start and end: Chromosome coordinates

context: Sequence context of cytosine i.e CG,CHG,CHH

posteriorMax: Posterior value of the methylation state call

status: Methylation status

rc.meth.lvl: Recalibrated methylation level calculated from the posteriors and fitted parameters

3 Generate DMR matrix

3.1 Run "makeDMRmatrix"

This function generates a DMR matrix of state calls, rc.meth.lvls and posterior probabilities for all samples in one dataframe.

samplelist: DATAFRAME OBJECT containing information about file, sample and replicate. For control/treatment data an additional column specifying the replicates is required.

input.dir: PATH to directory containing region files.

out.dir: PATH to output directory.

contexts: sequence contexts of the cytosine. By default this option is set to c("CG", "CHG", "CHH", "CHH"). If you want to run for a single context such as CG, set it as "CG".

postMax.out: By default this option is set as FALSE. You can set it to TRUE if you want to output the DMR matrix containing posterior probabilities for the status call of each region.

include.intermediate: LOGICAL specifying whether or not the intermediate component should be included in the HMM model. By default this option is set as FALSE.

```
makeDMRmatrix(contexts = c("CG", "CHG", "CHH"), postMax.out = TRUE,
              samplelist = listFiles1,
              input.dir = "./population",
              out.dir = "./population/matrix",
              include.intermediate = TRUE)

makeDMRmatrix(contexts = c("CG", "CHG", "CHH"), postMax.out = TRUE,
              samplelist = listFiles2,
              input.dir = "./replication",
              out.dir = "./replication/matrix",
              include.intermediate = TRUE)
```

3.2 Output files of makeDMRmatrix function

"CG_StateCalls.txt" has the following structure. "0" in the output matrix denotes "Unmethylated", "1" stands for "Methylated" and "0.5" stands for "Intermediate".

```
statecalls <- fread("./population/matrix/CG_StateCalls.txt", header = T)
head(statecalls)
```

```
##    seqnames start   end methylomeA_rep1 methylomeB_rep1 methylomeC_rep1
##       <int> <int> <int>           <num>           <num>           <num>
## 1:        1     1   200               0               1             0.5
## 2:        1    51   250               0               1             0.5
## 3:        1   101   300               0               1             0.5
## 4:        1   151   350               1               1             0.0
## 5:        1   201   400               1               0             0.0
## 6:        1   251   450               1               0             0.0
##    methylomeD_rep1
##              <num>
## 1:               1
## 2:               1
```

```
## 3:               1
## 4:               0
## 5:               0
## 6:               0
```

"CG_rcMethlvl.txt" has the following structure. The output matrix contains recalibrated methylation levels for each sample and for the specific region.

```
rcmethlvls <- fread("./population/matrix/CG_rcMethlvl.txt", header = T)
head(rcmethlvls)
```

```
##    seqnames start   end methylomeA_rep1 methylomeB_rep1 methylomeC_rep1
##       <int> <int> <int>           <num>           <num>           <num>
## 1:        1     1   200         0.04022         0.91114         0.48309
## 2:        1    51   250         0.12897         0.85744         0.54839
## 3:        1   101   300         0.12897         0.85744         0.54839
## 4:        1   151   350         0.80224         0.85744         0.47546
## 5:        1   201   400         0.80224         0.47885         0.47546
## 6:        1   251   450         0.80224         0.47885         0.47546
##    methylomeD_rep1
##              <num>
## 1:         0.90956
## 2:         0.79152
## 3:         0.79152
## 4:         0.49508
## 5:         0.49508
## 6:         0.49508
```

"CG_postMax.txt" has the following structure. The output matrix contains posterior probabilities for each sample and for the specific region.

```
postMax <- fread("./population/matrix/CG_postMax.txt", header = T)
head(postMax)
```

```
##    seqnames start   end methylomeA_rep1 methylomeB_rep1 methylomeC_rep1
##       <int> <int> <int>           <num>           <num>           <num>
## 1:        1     1   200         0.97962         0.99733         0.98386
## 2:        1    51   250         0.79295         0.87356         0.82874
## 3:        1   101   300         0.79295         0.87356         0.82874
## 4:        1   151   350         0.64649         0.87356         0.33333
## 5:        1   201   400         0.64649         0.33333         0.33333
## 6:        1   251   450         0.64649         0.33333         0.33333
##    methylomeD_rep1
##              <num>
## 1:         0.88337
## 2:         0.64923
## 3:         0.64923
## 4:         0.33333
## 5:         0.33333
## 6:         0.33333
```

3.3 Split DMR matrix into pairwise groups

Ignore this step if you are running jDMR on population data without replicates. For pairwise control-treatment data-sets with replicates, you need to run this step. This function generates a DMR matrix of state calls, rc.meth.lvls and posterior probabilities for each pairwise control-treatment in one dataframe. For example, if the samplelist includes control, treatment1 and treatment2, this function generates a DMR matrix of state calls, rc.meth.lvls and posterior probabilities for control-treatment1 in one dataframe, and a DMR matrix of state calls, rc.meth.lvls and posterior probabilities for control-treatment2 in one dataframe.

samplelist: DATAFRAME OBJECT containing information about file, sample and replicate. For control/treatment data an additional column specifying the replicates is required.

input.dir: PATH to directory containing region files.

out.dir: PATH to output directory.

contexts: sequence contexts of the cytosine. By default this option is set to c("CG", "CHG", "CHH"). If you want to run for a single context such as CG, set it as "CG".

postMax.out: by default this option is set to FALSE. If you want to output the matrix containing posterior probabilities set it to TRUE.

```
splitGroups(samplelist = listFiles2, postMax.out = TRUE, contexts = c("CG", "CHG", "CHH"),
            input.dir = "./replication/matrix",
            out.dir = "./replication/matrix")
```

3.4 Output files of splitGroups function

```
statecalls_split <- fread("./replication/matrix/CG_WT_mutant1_StateCalls.txt", header = T)
head(statecalls_split)
```

```
##    seqnames start   end WT_rep1 WT_rep2 mutant1_rep1 mutant1_rep2
##       <int> <int> <int>   <num>   <num>        <num>        <num>
## 1:        1     1   200       0       1          0.5            1
## 2:        1    51   250       0       1          0.5            1
## 3:        1   101   300       0       1          0.5            1
## 4:        1   151   350       1       1          0.0            0
## 5:        1   201   400       1       0          0.0            0
## 6:        1   251   450       1       0          0.0            0
```

4 Filter DMR matrix

4.1 Filter the DMR matrix

This function filters the DMR matrix for non-polymorphic patterns.

For population data-sets without replicates, this function will find the outputs of 'runjDMRgrid' for filtering. For pairwise control-treatment data-sets with replicates, this function will find the outputs of 'splitGroups' for filtering.

data.dir: PATH to folder containing DMR matrix

epiMAF.cutoff: Applicable for calling calling population DMRs. This option can be used to filter for Minor Epi-Allele frequency as specified by user. By default, this option is set to NULL.

replicate.consensus: Applicable for control-treatment data-sets with replicates. Users can specify the percentage of concordance in methylation states in samples with multiple replicates. For datasets with just 2 replicates, replicate.consensus should be set as 1 (means 100% concordance). By default, this option is set to NULL.

samplelist: DATAFRAME OBJECT containing information about file, sample and replicate. For control/treatment data an additional column specifying the replicates is required.

if.mergingBins: Logical argument if merging consecutive bins having the same stateCalls should be performed. By default set to TRUE. (logical)

```
filterDMRmatrix(epiMAF.cutoff = 0.33, replicate.consensus = NULL,
                data.dir = "./population/matrix",
                samplelist = listFiles1, if.mergingBins = FALSE)
```

```
filterDMRmatrix(epiMAF.cutoff = NULL, replicate.consensus = 0.5,
                data.dir = "./replication/matrix",
                samplelist = listFiles2, if.mergingBins = FALSE)
```

4.2 Filtered Output

"CG_StateCalls-filtered.txt" has the following structure.

```
statecalls_filtered <- fread("./population/matrix/CG_StateCalls-filtered.txt", header = T)
head(statecalls_filtered)
```

```
##    seqnames start   end methylomeA_rep1 methylomeB_rep1 methylomeC_rep1
##       <int> <int> <int>           <num>           <num>           <num>
## 1:        1     1   200               0               1             0.5
## 2:        1    51   250               0               1             0.5
## 3:        1   101   300               0               1             0.5
## 4:        1   151   350               1               1             0.0
## 5:        1   201   400               1               0             0.0
## 6:        1   251   450               1               0             0.0
##    methylomeD_rep1
##              <num>
## 1:               1
## 2:               1
## 3:               1
## 4:               0
```

```
## 5:               0
## 6:               0
```

```
statecalls_filtered <- fread("./replication/matrix/CG_WT_mutant1_StateCalls-filtered.txt", header = T)
head(statecalls_filtered)
```

```
##    seqnames start   end WT_rep1 WT_rep2 mutant1_rep1 mutant1_rep2
##       <int> <int> <int>   <num>   <num>        <num>        <num>
## 1:        1   151   350       1       1          0.0            0
## 2:        1   201   400       1       0          0.0            0
## 3:        1   251   450       1       0          0.0            0
## 4:        1   301   500       1       0          0.0            0
## 5:        1   351   550       0       0          0.5            1
## 6:        1   401   600       0       0          0.5            1
```

5 Search for context-specific and annotate DMRs

5.1 Output context specific DMR

Output DMRs specific for contexts i.e CG-only, CHG-only, CHH-only, non-CG and multi-context DMRs using the StateCalls_filtered.txt files (if variable if Filtered equals to TRUE) or StateCalls.txt files (if variable ifFiltered equals to FALSE, as default).

```
context.specific.DMRs(samplelist = listFiles2,
                      output.dir="./replication/context_DMRs",
                      input.dir = "./replication/matrix",
                      if.filtered = TRUE)
```

5.2 Annotate DMRs

This function annotates the lists of DMRs. Any file(.txt) containing 3 columns (chr, start, end) can be annotated using the annotateDMRs function. Please move all files to be annotated to a separate folder and set the full PATH to the "input.dir" option.

gff.files: Multiple gff3 annotation files can be supplied as a vector

annotation: specify annotation categories

input.dir: path to folder containing only files to be annotated. Any file containing 3 columns (chr, start, end) can be annotated using the annotateDMRs function.

if.gff3: whether to output annotated files in gff3 format

out.dir: path to output folder

In the following example, I will annotate the files generated in section 3.1

```
gff.file_promoters <- system.file("extdata/toyData", "TAIR10_promoters.gff3", package = "jDMRgrid")
gff.file_TE <- system.file("extdata/toyData", "TAIR10_TE.gff3", package = "jDMRgrid")

annotateDMRs(gff.files = c(gff.file_promoters, gff.file_TE),
             annotation = c("promoters", "TE"),
             input.dir = "./replication/annotation",
             if.gff3 = FALSE,
             out.dir = "./replication/annotation")
```

5.3 Output files after annotation

Mapped files are output in .txt and/or .gff3 format. Addiitonally, a DMR count table is generated.

```
annotations <- fread("./replication/annotation/CG_WT_mutant1_StateCalls_annotation.txt",
                     header = T)
head(annotations)
```

```
##    seqnames start   end          id
##       <int> <int> <int>      <char>
## 1:        1  1951  2150 AT1G01010_p
## 2:        1  2001  2200 AT1G01010_p
## 3:        1  2051  2250 AT1G01010_p
```

```
## 4:        1  2101  2300 AT1G01010_p
## 5:        1  2151  2350 AT1G01010_p
## 6:        1  2201  2400 AT1G01010_p
```

```
DMRcounts <- fread("./replication/annotation/DMR-counts.txt", header = T)
head(DMRcounts)
```

```
##                      sample total.DMRs promoters    TE multiple.overlaps
##                      <char>      <int>     <int> <int>             <int>
## 1: CG_WT_mutant1_StateCalls         56        21     0                 0
```

6 R session info

```
sessionInfo()
```

```
## R version 4.4.1 (2024-06-14)
## Platform: x86_64-pc-linux-gnu
## Running under: Ubuntu 22.04.5 LTS
##
## Matrix products: default
## BLAS:   /usr/lib/x86_64-linux-gnu/openblas-pthread/libblas.so.3
## LAPACK: /usr/lib/x86_64-linux-gnu/openblas-pthread/libopenblasp-r0.3.20.so;  LAPACK version 3.10.0
##
## 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
##
## time zone: Etc/UTC
## tzcode source: system (glibc)
##
## attached base packages:
## [1] stats     graphics  grDevices utils     datasets  methods   base
##
## other attached packages:
## [1] data.table_1.16.2 jDMRgrid_0.2.4
##
## loaded via a namespace (and not attached):
##  [1] tidyselect_1.2.1            dplyr_1.1.4
##  [3] R.utils_2.12.3              Biostrings_2.70.3
##  [5] bitops_1.0-9                fastmap_1.2.0
##  [7] RCurl_1.98-1.16             GenomicAlignments_1.40.0
##  [9] XML_3.99-0.17               digest_0.6.37
## [11] lifecycle_1.0.4             magrittr_2.0.3
## [13] compiler_4.4.1              rlang_1.1.4
## [15] tools_4.4.1                 utf8_1.2.4
## [17] yaml_2.3.10                 rtracklayer_1.64.0
## [19] knitr_1.48                  S4Arrays_1.4.1
## [21] curl_5.2.3                  DelayedArray_0.30.1
## [23] plyr_1.8.9                  abind_1.4-8
## [25] BiocParallel_1.38.0         withr_3.0.2
## [27] purrr_1.0.2                 R.oo_1.27.0
## [29] BiocGenerics_0.50.0         grid_4.4.1
## [31] stats4_4.4.1                fansi_1.0.6
## [33] colorspace_2.1-1            future_1.34.0
## [35] ggplot2_3.5.1               globals_0.16.3
## [37] scales_1.3.0                iterators_1.0.14
## [39] SummarizedExperiment_1.34.0 cli_3.6.3
## [41] rmarkdown_2.28              crayon_1.5.3
## [43] generics_0.1.3              rstudioapi_0.17.1
## [45] future.apply_1.11.3         httr_1.4.7
```

```
## [47] reshape2_1.4.4              rjson_0.2.23
## [49] RUnit_0.4.33                ape_5.8
## [51] stringr_1.5.1               zlibbioc_1.50.0
## [53] parallel_4.4.1              BiocManager_1.30.25
## [55] XVector_0.44.0              restfulr_0.0.15
## [57] matrixStats_1.4.1           vctrs_0.6.5
## [59] Matrix_1.7-1                jsonlite_1.8.9
## [61] minpack.lm_1.2-4            IRanges_2.38.1
## [63] S4Vectors_0.42.1            RBGL_1.80.0
## [65] listenv_0.9.1               foreach_1.5.2
## [67] tidyr_1.3.1                 glue_1.8.0
## [69] parallelly_1.38.0           codetools_0.2-19
## [71] stringi_1.8.4               gtable_0.3.6
## [73] GenomeInfoDb_1.40.1         GenomicRanges_1.56.2
## [75] BiocIO_1.14.0               UCSC.utils_1.0.0
## [77] munsell_0.5.1               tibble_3.2.1
## [79] pillar_1.9.0                htmltools_0.5.8.1
## [81] graph_1.82.0                methimpute_1.26.0
## [83] GenomeInfoDbData_1.2.12     R6_2.5.1
## [85] doParallel_1.0.17           lattice_0.22-5
## [87] evaluate_1.0.1              Biobase_2.64.0
## [89] R.methodsS3_1.8.2           Rsamtools_2.20.0
## [91] Rcpp_1.0.13-1               nlme_3.1-165
## [93] SparseArray_1.4.8           xfun_0.49
## [95] MatrixGenerics_1.16.0       biocViews_1.72.0
## [97] pkgconfig_2.0.3
```