MMDiff2: Statistical Testing for ChIP-Seq Data Sets

Gabriele Schwikert <`G.Schweikert@ed.ac.uk`> [1] ***and David Kuo*** <`dkuo@cbio.mskcc.org`> [2]

[1em] [1] The Informatics Forum, University of Edinburgh and
The Wellcome Trust Center for Cell Biology, University of Edinburgh;
[2] Department of Physiology, Biophysics and Systems Biology, Weill Cornell Medical College and
Computational Biology Department, Memorial Sloan Kettering Cancer Center

Modified: 25 Mar, 2016. Compiled: November 24, 2026

Contents

1 Introduction . . . 1
2 Quick Start . . . 2
3 Analysis Pipeline . . . 3
3.1 Loading Data . . . 3
3.2 Processing Peaks from BAM Files . . . 3
3.3 Examining the `DBAmmd` Object and Reporting . . . 4
3.4 Plotting Peaks . . . 5
3.4.1 Motif Enrichment within Peaks . . . 5
3.4.2 Gene Annotation . . . 6
3.5 Computing per-region pair-wise distances between samples . . . 7
3.6 Differential Testing . . . 7
3.7 Analyzing Results . . . 7
3.8 Shiny Application Usage . . . 8
4 Comparison with DiffBind . . . 9
5 Setup . . . 9

1 Introduction

ChIP-seq is the standard technique for determining genome-wide protein-DNA interactions and of particular importance for the fields of regulatory biology and epigenetics. A common use-case for ChIP-seq is determining where transcription factors bind on the genome. The resulting ChIP-seq peaks for transcription factors are typically *narrow* and can be further

queried for known DNA-binding motifs. Another example is ascertaining chromatin states by profiling histone modifications. Specific post-translational modification on histone proteins, represented as *broad* peaks, are highly informative of the gene activity and regulatory status.

The utility and reproducibility of ChIP-seq led to its use in large-scale projects such as *ENCODE* and *Epigenomics Roadmap*. Thousands of ChIP-seq datasets were generated for various families of transcription factors and histone modifications in cell lines and primary tissues. These projects described the regulatory landscape through the lens of ChIP-seq experiments and also established a standard for comparison with other experimental datasets.

With an abundance of ChIP-seq data, there have also emerged many different computational tools for the analysis and statistical testing of ChIP-seq data between conditions, iė; control cells against experimentally treated cells. After mapping reads to the genome, ChIP-seq analysis begins by calling peaks between a sample conjugated with an antibody against a control with DNA input. Various computational tools exist for this peak calling task but a popular tool is *MACS2*. In another scenario, regions of interest are pre-defined with regard to other annotated objects, for example by applying windows around transcription start sites (TSS). In querying all of the TSS for possible ChIP-seq reads, experimentalists can observe the density of reads near regions of interest.

After regions/peaks are identified, a common analysis is determining regions which are differentially bound by the studied transcription factor, or which peaks demonstrate differential histone modifications between conditions. For simplicity, we will call these regions differential regions. Often this is treated as a read-count based test, and some statistical packages such as `DESeq`2 can be utilized for this task. ChIP-seq specific tools such as `DiffBind`also integrate count-based testing in their methods.

A caveat of these methods is that count-based methods disregard higher order features of ChIP-seq reads such as peak shape. Between two conditions, differential read density may not be significant but the distribution of reads within the peak may imply differential protein binding or the presence of protein complexes. Detection of these differential peak shapes is not possible when simplifying peaks to single values, as is the case for the aforementioned read-count based tests.

We therefore incorporate higher order ChIP-seq peak information into a test for differential binding by adapting kernel-based statistical tests to ChIP-Seq data. This package was initially released as *MMDiff*.

In this current release, *MMDiff2*, we improve code stability, decrease analysis runtime and provide an interactive Shiny application for exploring peak profiles.

2 Quick Start

```
library('MMDiff2')
library('MMDiffBamSubset')
ExperimentData <- list(genome='BSgenome.Mmusculus.UCSC.mm9',
                       dataDir=system.file("extdata", package="MMDiffBamSubset"),
                       sampleSheet="Cfp1.csv")
MetaData <- list('ExpData' = ExperimentData)
MMD <- DBAmmd(MetaData)
data("Cfp1-Peaks")
MMD <- setRegions(MMD,Peaks)
MMD <- getPeakReads(MMD)
```

```
MMD <- estimateFragmentCenters(MMD)
MMD <- compHists(MMD)
MMD <- compDists(MMD)
MMD <- setContrast(MMD,contrast='byCondition')
MMD <- compPvals(MMD)
res <- reportResults(MMD)
```

3 Analysis Pipeline

`MMDiff2`requires position sorted and indexed `BAM` files of the sample and input libraries and a set of regions in `GRanges` format, for example `MACS2` called peaks. For this vignette, we utilize the `MMDiffBamSubset` dataset that contains ChIP-seq peaks and reads from Clouaire *et al.*, Genes & Dev. 2012.

```
# load software package
library('MMDiff2')
```

```
## Warning in fun(libname, pkgname):  Package 'MMDiff2' is deprecated and will be
removed from
##   Bioconductor version 3.25
```

3.1 Loading Data

We load the `MMDiffBamSubset` data and specify an experimental genome (mm9), a directory containing peaks and reads, and a *csv* file that details the paths to different samples. The `MetaData` object will be called upon later for adding binding motif and gene annotation information.

```
# load data packages
library('MMDiffBamSubset')
```

```
## Warning in fun(libname, pkgname):  Package 'MMDiffBamSubset' is deprecated and
will be removed from
##   Bioconductor version 3.25
```

```
# create metaData:

ExperimentData <- list(genome = 'BSgenome.Mmusculus.UCSC.mm9',
                    dataDir = system.file("extdata", package="MMDiffBamSubset"),
                    sampleSheet="Cfp1.csv")

MetaData <- list('ExpData' = ExperimentData)
```

3.2 Processing Peaks from BAM Files

A `GenomicRanges` object that summarizes the peaks has been provided in this vignette. After loading this file, we instantiate the `DBAmmd` class and read the `BAM` files for all peaks in all conditions.

```
data('Cfp1-Peaks')
MMD <- DBAmmd(MetaData)
MMD <- setRegions(MMD,Peaks)
MMD <- getPeakReads(MMD)
```

Note that we are fetching the exact start and end positions of mapped fragments and not the coverage. In the case of single-end reads, the left-most positions of fragments mapping to the positive strand are stored in a list called `Left.p`, and the right most positions of fragments mapping to the negative strand are stored in `Right.n`. For paired-end reads, `Right.p` and `Left.n` are additionally kept.

The next step in the pipeline is to estimate the fragment centers using information from the forward and reverse strands. For paired-end reads, we compute the exact Center position for each fragment. For single-strand libraries, we estimate the Centers. We improve the resolution of the estimated Center positions, relative to coverage-based approach, by utilizing the method described in Gomes *et al.* Genome Research, 2014.

```
MMD <- estimateFragmentCenters(MMD)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##     0.0   140.0   160.0   155.4   180.0   230.0
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##    50.0   130.0   160.0   154.5   190.0   240.0
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##    40.0   152.5   170.0   169.4   200.0   260.0
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
## -360.00   30.00   90.00   98.71  145.00  400.00
```

Once the fragment centers have been estimated, histograms for each peak are computed using the read information. The step is essential for plotting regions and not essential in order to determine differentially shaped regions. Two parameters as used to compute histograms: `bin.length`, which determines the smoothing parameter over the set of reads (Default: 20) and the
whichPos, which determines the fragment position from which histograms should be computed: `'Left.p'` for left ends of fragments mapping to positive strand, `'Right.n'` for right ends of fragments mapping to negative strands and `'Center'` for the Fragment centers for all positions (Default: `Center`).

```
MMD <- compHists(MMD, bin.length=20, whichPos="Center")
```

3.3 Examining the `DBAmmd` Object and Reporting

After peak histograms have been computed, we have a complete `DBAmmd` object. This object contains several components including:

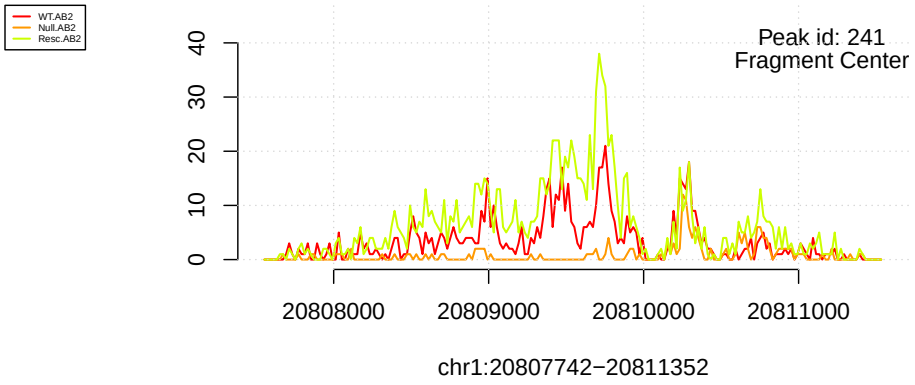
1. Peaks
2. Samples
3. Distances between Conditions (not yet computed)
4. Distances within Conditions (not yet computed)
5. P-values (not yet computed)

Each of these components will be highlighted in the subsequent sections.

3.4 Plotting Peaks

Peaks can be visualized by providing a `Peak.id` parameter to the `plotPeak` function. Line plots are drawn that signify the different histograms from each of the samples.

```
plotPeak(MMD, Peak.id='241', plot.input = FALSE, whichPos="Center")
```



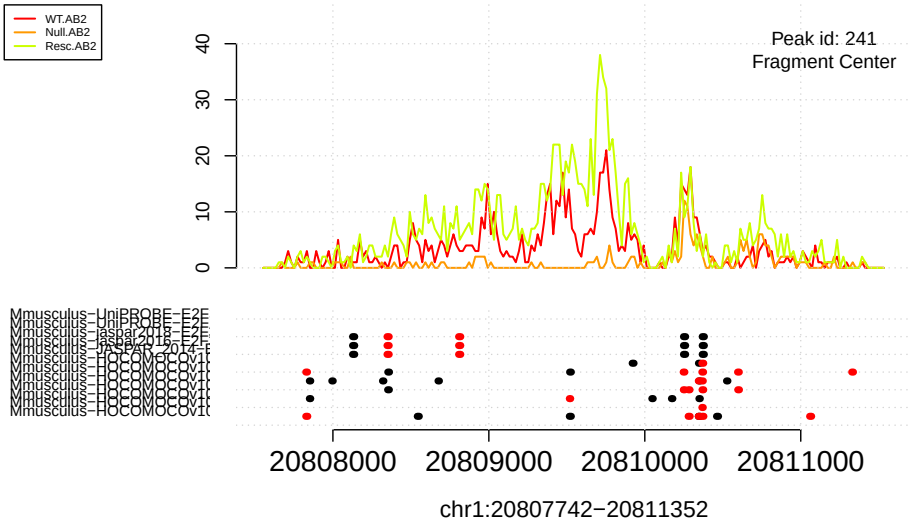
3.4.1 Motif Enrichment within Peaks

An additional feature to `MMDiff2`is plotting motifs within peaks. We specify motifs with the following command against the `MotifDb`:

```
library('MotifDb')

## See system.file("LICENSE", package="MotifDb") for use restrictions.

motifs <- query(query(MotifDb, 'Mmusculus'), 'E2F')
```

The query will return motifs that partially match the term "Pax", iė; Pax4 and Pax5 could be plotted. Integration of the motifs is seamless into the `plotPeak` function:

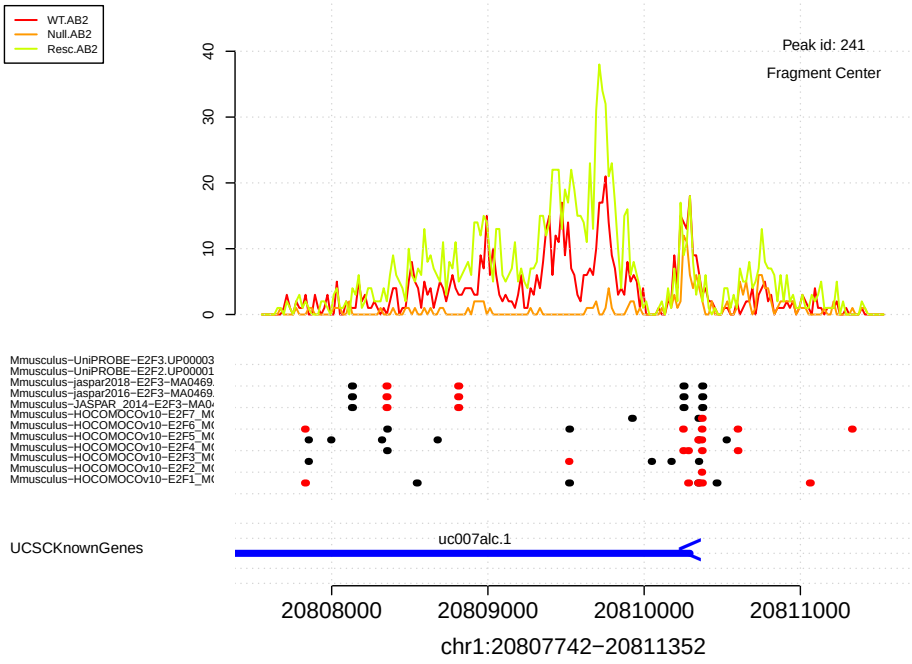
```
plotPeak(MMD, Peak.id='241', NormMethod=NULL,plot.input = FALSE,whichPos="Center",
  Motifs=motifs,Motifcutoff="80%")
```



3.4.2 Gene Annotation

In addition to plotting motifs of interest, we have enabled in `MMDiff2`gene annotation tracks that can also be plotted in a fully integrated plot with the ChIP-seq peak by providing a `genomicRanges` object to the `anno` parameter.

```
data("mm9-Genes")
names(GR) <- GR$tx_name
GR <- list(UCSCKnownGenes = GR)
plotPeak(MMD, Peak.id='241', NormMethod=NULL,plot.input = FALSE,
         whichPos="Center",Motifs=motifs, anno=GR)
```



3.5 Computing per-region pair-wise distances between samples

The core of `MMDiff2`is the comparison of peak shapes between samples. This is performed by using a kernel-based test that calculates 1) a distance between treatment conditions and 2) within conditions. The `compDists` function generates these per-region and pair-wise distances for all samples in the experiment.

3.6 Differential Testing

After distances are generated, `MMDiff2`computes p-values based on the specified conditions.

```
MMD <- setContrast(MMD,contrast='byCondition')
```

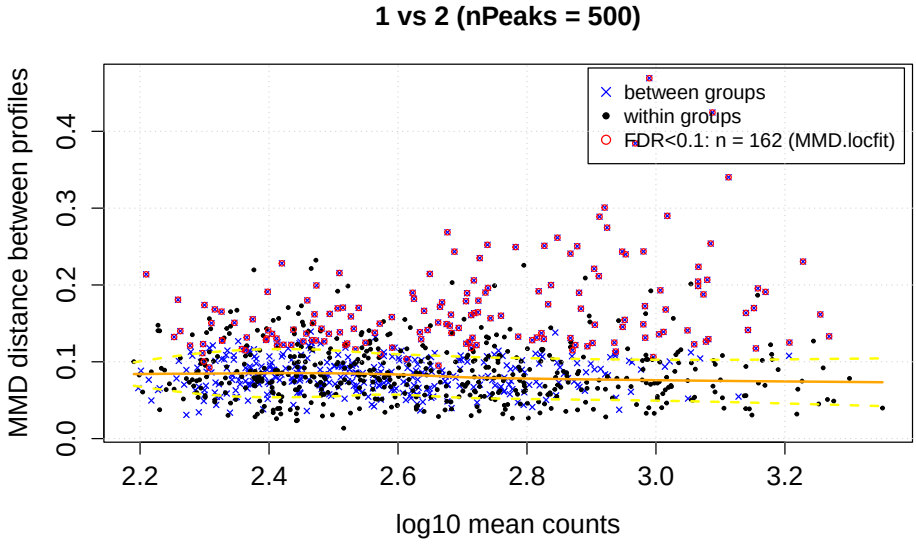
The contrast can also be manually set:

```
MMD <- compPvals(MMD,dist.method='MMD')

## Warning in compPvals(MMD, dist.method = "MMD"):  no replicates given for group2

## Computing p-values
```

The `plotDists` function plots a summary of all of the peak distances 1) between groups and 2) within groups. Significant differentially shaped peaks are highlighted in red.

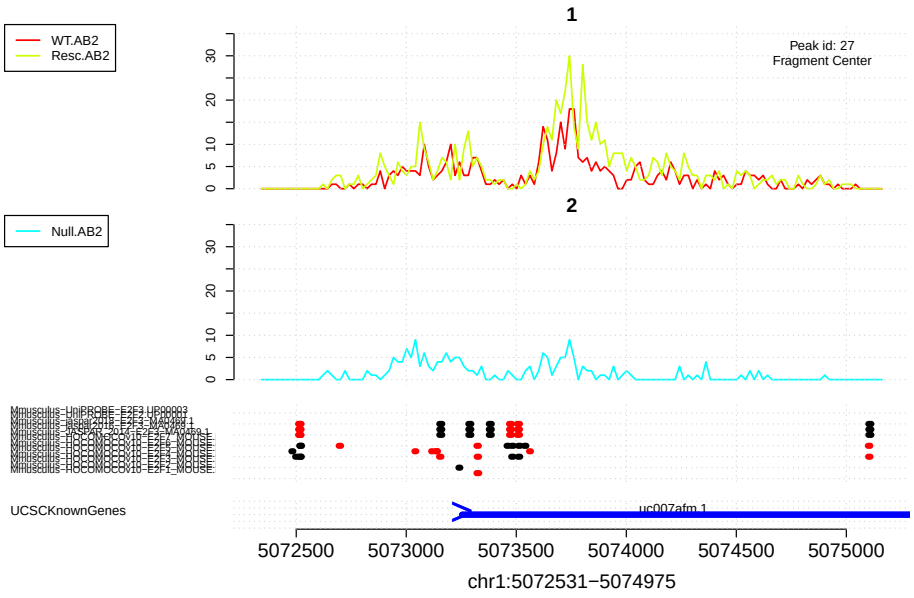
3.7 Analyzing Results

```
plotDists(MMD, dist.method='MMD',whichContrast=1,
                           diff.method='MMD.locfit',
                           bUsePval=FALSE, th=0.1,
                           title=NULL, what=3,
                           xlim=NULL,ylim=NULL,Peak.IDs=NULL,
                           withLegend=TRUE)
```




```
res <- reportResults(MMD)
```

```
Peak.ids <- names(res)

plotPeak(MMD, Peak.id=Peak.ids[1], NormMethod=NULL,plot.input = FALSE,
         whichPos="Center",Motifs=motifs, anno=GR,whichContrast = 1)

## No Samples specified, plotting all samples

## No normalization factors applied

## checking availability of package BSgenome.Mmusculus.UCSC.mm9
```




```
dev.off()

## null device
##           1

plotDISTS4Peak(MMD,Peak.id=Peak.ids[1],dist.method='MMD',
               whichContrast=1,Zoom=TRUE)
```

3.8 Shiny Application Usage

`MMDiff2` features an interactive Shiny application that displays distances plots showing all peaks and in-depth peak plots.

Beginning in the top panels of the Shiny application are two `plotDists` outputs. In the left panel, drawing a box dynamically zooms in on the right panel. The drawn box can also be panned around the plot. To remove the zoomed view, click once outside the box.

On the right panel, clicking anywhere on the plot will find the nearest peak. The nearest peak to the mouse-click will then generate two plots below.

The bottom left panel is a histogram representation of all conditions (`plotPeak`) while the bottom right panel displays the distances between samples for that peak (`plotDISTS4peak`).

Finally, a UCSC genome browser link is generated that matches the coordinates of the selected peak for additional inspection.

```
runShinyMMDiff2(MMD)
```

A demonstration of this is shown below.



UCSC Link:
http://genome.ucsc.edu/cgi-bin/hgTracks?org=mouse&db=mm10&position=chr1:31003101-31006284

4 Comparison with DiffBind

To compare against the `DiffBind` package, we downloaded aligned `BAM` files from the Ross-Innes *et al.* 2012 paper and used the same called peaks as analyzed in the *DiffBind* vignette.

5 Setup

This vignette was built using:

```
sessionInfo()
```

```
## R version 4.6.1 (2026-06-24)
## Platform: x86_64-pc-linux-gnu
## Running under: Ubuntu 24.04.5 LTS
##
## Matrix products: default
## BLAS:   /home/biocbuild/bbs-3.24-bioc/R/lib/libRblas.so
## LAPACK: /usr/lib/x86_64-linux-gnu/lapack/liblapack.so.3.12.0  LAPACK version 3.12.0
##
```

```
## locale:
##  [1] LC_CTYPE=en_US.UTF-8       LC_NUMERIC=C
##  [3] LC_TIME=en_GB              LC_COLLATE=C
##  [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: America/New_York
## tzcode source: system (glibc)
##
## attached base packages:
## [1] stats4    stats     graphics  grDevices utils     datasets  methods
## [8] base
##
## other attached packages:
##  [1] BSgenome.Mmusculus.UCSC.mm9_1.4.0 BSgenome_1.82.0
##  [3] rtracklayer_1.74.0                BiocIO_1.24.0
##  [5] MotifDb_1.56.0                    MMDiffBamSubset_1.49.1
##  [7] MMDiff2_1.42.0                    Biobase_2.74.0
##  [9] Rsamtools_2.30.0                  Biostrings_2.82.0
## [11] XVector_0.54.0                    GenomicRanges_1.66.0
## [13] IRanges_2.48.0                    S4Vectors_0.52.0
## [15] Seqinfo_1.4.0                     BiocGenerics_0.60.0
## [17] generics_0.1.4
##
## loaded via a namespace (and not attached):
##  [1] gtable_0.3.6             rjson_0.2.23
##  [3] xfun_0.62                ggplot2_4.0.4
##  [5] lattice_0.23-1           vctrs_0.7.3
##  [7] tools_4.6.1              bitops_1.1-0
##  [9] curl_8.2.0               parallel_4.6.1
## [11] tibble_3.3.1             highr_0.12
## [13] pkgconfig_2.0.3          BiocBaseUtils_1.16.0
## [15] data.table_1.18.6.1      RColorBrewer_1.1-3
## [17] S7_1.0.0                 cigarillo_1.4.0
## [19] lifecycle_1.0.5          compiler_4.6.1
## [21] farver_2.1.2             tinytex_0.61
## [23] BiocStyle_2.42.0         codetools_0.2-20
## [25] httpuv_1.6.17            htmltools_0.5.9
## [27] RCurl_1.98-1.20          yaml_2.3.12
## [29] pillar_1.11.1            later_1.4.8
## [31] crayon_1.5.3             BiocParallel_1.48.0
## [33] mime_0.13                tidyselect_1.2.1
## [35] locfit_1.5-9.12          digest_0.6.39
## [37] dplyr_1.2.1              restfulr_0.0.17
## [39] fastmap_1.2.0            grid_4.6.1
## [41] cli_3.6.6                magrittr_2.0.5
## [43] dichromat_2.0-1          XML_3.99-0.25
## [45] scales_1.4.0             promises_1.5.0
## [47] rmarkdown_2.32           httr_1.4.9
```

```
## [49] matrixStats_1.5.0        otel_0.3.0
## [51] shiny_1.15.0             evaluate_1.0.5
## [53] knitr_1.52               rlang_1.3.0
## [55] Rcpp_1.1.2               xtable_1.8-8
## [57] glue_1.8.1               BiocManager_1.30.27
## [59] splitstackshape_1.4.8.1  R6_2.6.1
## [61] GenomicAlignments_1.50.0
```