MEDIPS Tutorial

Lukas Chavez

April 10, 2013

Contents

1 Introduction

MEDIPS was developed for analyzing data derived from methylated DNA immunoprecipitation (MeDIP) experiments [Weber et al., 2005] followed by sequencing (MeDIP-seq) [Down et al., 2008]. However, MEDIPS provides several functionalities for the analysis of other kinds of quantitative sequencing data

(e.g. ChIP-seq, MBD-seq, CMS-seq, and others) including calculation of differential coverage between groups of samples as well as saturation and correlation analyses.

In detail, MEDIPS adresses the following aspects in the context of quantitative sequencing data analysis:

- estimating the reproducibility for obtaining full genome short read coverage profiles (saturation analysis),
- calculating genome wide signal densities at a user specified resolution (counts, rpkm),
- calculating differential coverage comparing two groups of samples,
- correcting for copy number variations present in the genomic background of the samples based on Input samples (if available),
- export of raw and normalized data as Wiggle files for visualization in common genome browsers (e.g. the UCSC genome browser).

In addition, MEDIPS provides the following MeDIP/MBD-seq specific functionalities:

- analyzing the coverage of genome wide DNA sequence patterns (e.g. CpGs) by the given short reads (or their mapping results, respectively),
- calculating a CpG enrichment factor as a quality control for MeDIP/MBD specific immunoprecipitation,
- calculating genome wide sequence pattern densities (e.g. CpGs) at a user specified resolution,
- plotting of calibration plots as a data quality check and for a visual inspection of the dependency between local sequence pattern (e.g. CpG) densities and MeDIP/MBD signals,
- normalization of MeDIP-seq data with respect to local sequence pattern (e.g. CpG) densities (relative methylation score),

MEDIPS starts where the mapping tools stop (typically bam files) and can be used for any genome of interest. In case a genome of interest is not available as a `BSgenome` package but the sequence of the genome is available, a custom `BSgenome` package can be generated, please see the "How to forge a BSgenome data package" manual of the `BSgenome` package.

2 Updates

Except of several bug fixes, there was no major update of the MEDIPS package since its release mid of 2010. With the release of version 1.10.0, there are several conceptual updates which result in major changes concerning data input, function and parameter definitions, and result formats. Hence, we apologize for any inconveniences, but we believe that these updates were necessary with respect to methodology, functionality, and performance (speed and memory usage).

Major updates include:

- MEDIPS allows for including replicates. For each group (control, treatment, control input, and treatment input) an arbitrary number of replicates can be provided.
- The concept of smaller bins within genome wide windows is replaced by counting short read coverage directly at genome wide windows. Normalization and differential coverage is then calculated window-wise.
- Genome wide differential coverage between two groups of samples can be calculated by modelling a negative binomial distribution. For this, MEDIPS applies the sophisticated Bioconductor package edgeR [MD et al., 2010] and its functions for normalization, estimation of overdispersion, and statistical testing. Therefore, MEDIPS not only allows to calculate differentially methylated regions from MeDIP-seq data but also allows to process many other kinds of quantitative sequencing data (e.g. ChIP-seq) in order to calculate genome wide differential coverage. Differential coverage can then be interpreted as either differential methylation (MeDIP/MBD or ChIP for histone modifications), differential hydroxymethylation (anti-CMS), differential binding (ChIP for transcription factors) or simply as sample specific enrichments (e.g. ChIP vs. Input).
- MEDIPS allows for estimating copy number variations (CNVs) when Input data for both conditions (Input control and Input treatment) is provided. For this, MEDIPS applies the Bioconductor package DNAcopy [Seshan and Olshen]. Subsequently, genomic regions showing differential coverage in the IP samples can be corrected for copy number variations present in the genomic background of the samples.
- MEDIPS accepts bam files as input.
- MEDIPS accepts fixedStep wiggle files as input.

3 Installation

In order to execute MEDIPS, you need to have some other packages installed in your R library. These are `BSgenome`, `edgeR`, `DNAcopy`, `Rsamtools`, `rtracklayer`, and `gtools`, as well as the packages they depend on. You can check this, by starting R and typing

```
> packageDescription("BSgenome")
> packageDescription("gtools")
> packageDescription("edgeR")
> packageDescription("DNAcopy")
> packageDescription("Rsamtools")
> packageDescription("rtracklayer")
```

R will inform you in case you do not have these packages installed. In case you do not have these necessary packages installed, start R and try typing

```
> source("http://bioconductor.org/biocLite.R")
> biocLite("BSgenome")
> biocLite("gtools")
```

```
> biocLite("edgeR")
> biocLite("DNAcopy")
> biocLite("Rsamtools")
> biocLite("rtracklayer")
```

Next, it is necessary to install the MEDIPS package into your R environment. Start R and enter:

```
> source("http://bioconductor.org/biocLite.R")
> biocLite("MEDIPS")
```

Next, it is necessary to have the genome of interest available in your R environment. As soon as you have the BSgenome package installed and the library loaded using

```
> library("BSgenome")
```

you can list all available genomes by typing

```
> available.genomes()
```

In the given example, we mapped the short reads against the human genome build hg19. Therefore, we download and install this genome build:

```
> source("http://bioconductor.org/biocLite.R")
> biocLite("BSgenome.Hsapiens.UCSC.hg19")
```

This takes some time, but has to be done only once for each reference genome.

Finally, the MEDIPS workflow described below requires access to example data available in the `MEDIPSData` package which can be installed by typing:

```
> source("http://bioconductor.org/biocLite.R")
> biocLite("MEDIPSData")
```

4 Preparations

First, the MEDIPS package has to be loaded. Herewith, the dependent libraries `BSgenome`, `Rsamtools`, `edgeR`, `DNAcopy`, `gtools`, `rtracklayer` as well as the packages they depend on will be loaded.

```
> library(MEDIPS)
```

In the given example, we mapped the short reads against the human genome build hg19. Therefore, we load the pre-installed (see chapter 3) hg19 library:

```
> library(BSgenome.Hsapiens.UCSC.hg19)
```

MEDIPS requires mapping results in bam format as input, which is a standard format for mapping results. In addition, MEDIPS allows for mapping results in tab-separated (|) BED text files (chr | start | end | name | score | strand). MEDIPS can also directly read the precomputed genome coverage from fixedStep wiggle files. In this case, all specified chromosomes have to be described completely.

In order to present a typical MEDIPS workflow, we access BAM files as well as preprocessed data included in the data package `MEDIPSData`. In order to load the library, please type:

```
> library("MEDIPSData")
```

The example data has been generated based on the following BAM files: `hESCs.MeDIP.Rep1.chr22.bam` (7.6M), `hESCs.MeDIP.Rep2.chr22.bam` (14M), `hESCs.MeDIP.Rep3.chr22.bam` (9.1M), `hESCs.Input.chr22.bam` (4.9M), `DE.MeDIP.Rep1.chr22.bam` (12M), `DE.MeDIP.Rep2.chr22.bam` (14M), `DE.MeDIP.Rep3.chr22.bam` (13M), and `DE.Input.chr22.bam` (11M)

These bam files are the bowtie [Langmead et al., 2009] mapping results of MeDIP-seq data derived from three replicates of human embryonic stem cells (hESCs) and replicates of differentiated hESCs (definitive endoderm, DE) [Chavez et al., 2010]. In addition, for each condition there is one bam file containg the mapping results of corresponding Input-seq data.

In this manual, we access some BAM files located in the `extdata` subdirectory of the MEDIPSData package. We have to point to the example BAM files as follows:

```
> bam.file.hESCs.Rep1.MeDIP = system.file("extdata", "hESCs.MeDIP.Rep1.chr22.bam",
+     package = "MEDIPSData")
> bam.file.hESCs.Input = system.file("extdata", "hESCs.Input.chr22.bam",
+     package = "MEDIPSData")
> bam.file.DE.Input = system.file("extdata", "DE.Input.chr22.bam",
+     package = "MEDIPSData")
```

Some other bam files have been already preprocessed and will be accessed as internally saved RData objects (see below).

Typically, you will directly specify your data file at the `file` parameter. It is also possible to specify the full path to your input file together with the file name.

Next, we define several parameters which will be used throughout the manual. The reference genome is hg19:

```
> BSgenome="BSgenome.Hsapiens.UCSC.hg19"
```

MEDIPS will replace all reads which map to exactly the same start and end positions on the same strand by only one representative:

```
> uniq=TRUE
```

All reads will be extended to a length of 300nt according to the given strand information:

```
> extend=300
```

As an alternative to the extend parameter, the `shift` parameter can be specified. Here, the reads are not extended but shifted by the specified number of nucleotides with respect to the given strand infomation. One of the two parameters `extend` or `shift` has to be 0.

```
> shift=0
```

The genome will be divided into adjacent windows of length 100nt and all further calculations (short read coverage, differential coverage between conditions etc.) will be calculated for each window.

```
> ws=100
```

In this manual, we are going to process only one chromosome (i.e. chr22) in order to exemplify a typical workflow. Therefore, we specify the `chr.select` parameter. Please note, the example bam files contain only data for chr22 anyway. Therefore, it is not necessary to explicitly specify this chromosome in this example.

```
> chr.select="chr22"
```

5 Quality controls

MEDIPS provides three different quality controls. In this section, the quality controls are demonstrated for the MeDIP-seq hESCs replicate `hESCs_Rep1_MeDIP.bam`.

5.1 Saturation analysis

The saturation analysis addresses the question, whether the given set of mapped reads is sufficient to generate a saturated and reproducible coverage profile of the reference genome. Only if there is a sufficient number of short reads, the resulting genome wide coverage profile will be reproducible by another independent set of a similar number of short reads. The saturation analysis is not specific for MeDIP-seq data and can be applied to other types of sequencing data like e.g. ChIP-seq.

```
> sr = MEDIPS.saturation(file = bam.file.hESCs.Rep1.MeDIP, BSgenome = BSgenome,
+     uniq = uniq, extend = extend, shift = shift, window_size = ws,
+     chr.select = chr.select, nit = 10, nrit = 1, empty_bins = TRUE,
+     rank = FALSE)
```

The saturation analysis divides the total set of available regions into two distinct random sets (`A` and `B`) of equal size. Both sets `A` and `B` are again divided into random subsets of equal size where the number of subsets is determined by the parameter `nit` (default=10). For each set, `A` and `B`, the saturation analysis iteratively selects an increasing number of subsets and calculates short read coverage at genome wide windows where the window sizes are defined by the `window_size` parameter. In each iteration step, the resulting genome wide coverages for the current subsets of `A` and `B` are compared using pearson correlation. As the number of considered reads increases during each iteration step, it is assumed that the resulting genome wide coverages become more similar, a dependency that is expressed by an increased correlation.

It has to be noted that the saturation analysis can be performed on two independent sets of short reads only. Therfore, a true saturation for one given sample can only be calculated for half of the available short reads. As it is of interest to examine the reproducibility for the total set of available short reads of the given sample, the saturation analysis is followed by an estimated saturation analysis. For the estimated saturation analysis, the full set of given regions is artifically doubled by considering each given region twice. Subsequently, the described saturation analysis is performed on the artificially doubled set of regions (see also Supplementary Methods in [Chavez et al., 2010]).

The results of the saturation and of the estimated saturation analysis can be viewed by typing

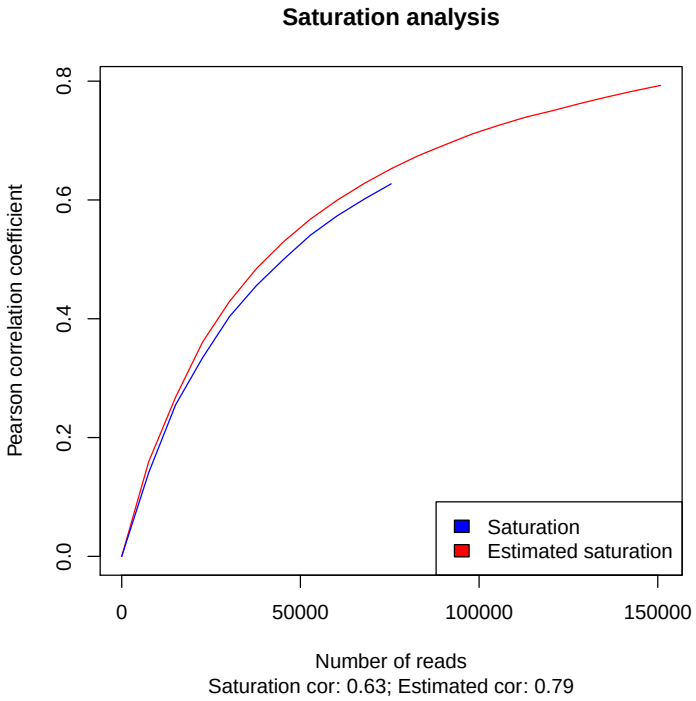
```
> sr
```

```
$distinctSets
       [,1]      [,2]
 [1,]     0 0.0000000
 [2,]  7539 0.1408847
 [3,] 15078 0.2551831
 [4,] 22617 0.3342984
 [5,] 30156 0.4036629
 [6,] 37695 0.4558762
 [7,] 45234 0.4997098
 [8,] 52773 0.5406283
 [9,] 60312 0.5733928
[10,] 67851 0.6013239
[11,] 75396 0.6271767

$estimation
        [,1]      [,2]
 [1,]      0 0.0000000
 [2,]   7539 0.1592750
 [3,]  15078 0.2680378
 [4,]  22617 0.3606441
 [5,]  30156 0.4291502
 [6,]  37695 0.4844322
 [7,]  45234 0.5294135
 [8,]  52773 0.5676084
 [9,]  60312 0.5996975
[10,]  67851 0.6277414
[11,]  75390 0.6525246
[12,]  82929 0.6744221
[13,]  90468 0.6928691
[14,]  98007 0.7109262
[15,] 105546 0.7257838
[16,] 113085 0.7394496
[17,] 120624 0.7503550
[18,] 128163 0.7622459
[19,] 135702 0.7732402
[20,] 143241 0.7834628
[21,] 150793 0.7928389

$numberReads
[1] 150793

$maxEstCor
[1] 1.507930e+05 7.928389e-01

$maxTruCor
[1] 7.539600e+04 6.271767e-01
```

The maximal obtained correlation of the saturation analysis is stored at the `maxTruCor` slot and the maximal obtained correlation of the estimated saturation analysis is stored at the `maxEstCor` slot of the saturation results object (here `sr`, first column: total number of considered reads, second column: correlation). The results of each iteration step are stored in the `distinctSets` and `estimation` slots for the saturation and estimated saturation analysis, respectively (first column: total number of considered reads, second column: obtained correlation).

In addition, the results can be visualized by typing

```
> MEDIPS.plotSaturation(sr)
```



Further parameters that can be specified for the saturation analysis are:

- `nrit`: methods that randomly select data entries may be processed several times in order to obtain more stable results. By specifying the `nrit` parameter (default=1) it is possible to run the saturation analysis several times. The final results returned to the saturation results object are the averaged results of all random iteration steps.

- `empty_bins`: can be either TRUE or FALSE (default TRUE). This parameter affects the way of calculating correlations between the resulting genome wide coverages. If there occur genomic bins which contain no overlapping regions, neither from the subset(s) of `A` nor from the subset(s) of `B`, these windows will be neglected when the paramter is set to FALSE.

- `rank`: can be either TRUE or FALSE (default FALSE). This parameter also effects the way of calculating correlations between the resulting

genome vectors. If `rank` is set to TRUE, the correlation will be calculated for the ranks of the bins instead of considering the counts (i.e. spearman correlation). Setting this parameter to TRUE is a more robust approach which reduces the effect of possible occuring outliers (i.e. windows with very high counts) on the correlation.

5.2 Sequence Pattern Coverage

The main idea of the sequence pattern coverage analysis is to test the number of CpGs (or any other predefined sequence pattern) covered by the given short reads. In addition, the depth of coverage per CpG is tested. The sequence pattern coverage analysis can be started by typing

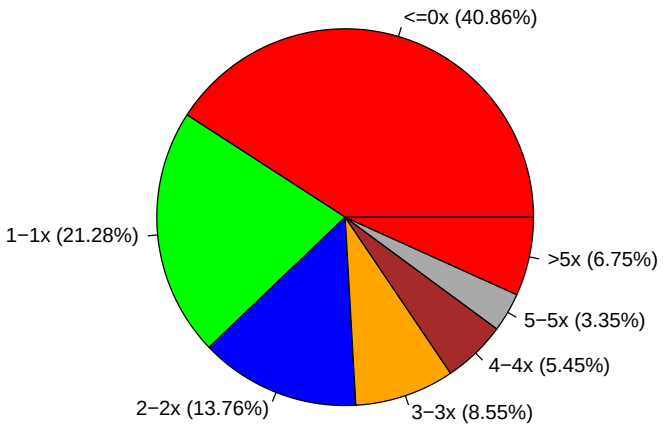
```
> cr = MEDIPS.seqCoverage(file = bam.file.hESCs.Rep1.MeDIP, pattern = "CG",
+     BSgenome = BSgenome, chr.select = chr.select, extend = extend,
+     shift = shift, uniq = uniq)
```

The sequence pattern coverage results object (here `cr`) contains four different slots:

- `cov.res`: is a vector of length equal to the total number of sequence patterns (e.g. CpGs). The entries correspond to the number of overlapping regions (coverage).
- `pattern`: the tested sequence pattern (e.g. CpG)
- `numberReads`: the total number of tested reads
- `numberReadsWO`: the number of reads which do not cover a tested sequence pattern

The results of the coverage analysis can be visualized by a pie chart:

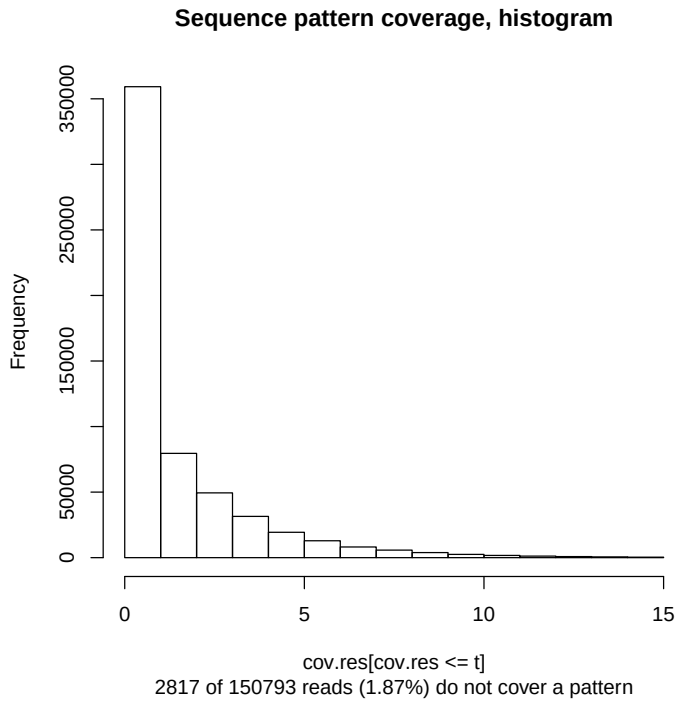
```
> MEDIPS.plotSeqCoverage(seqCoverageObj = cr, type = "pie", cov.level = c(0,
+     1, 2, 3, 4, 5))

Creating summary...
```



The pie chart illustrates the fraction of CpGs covered by the given reads according to their coverage level. The visualized coverage levels can be adjusted by the `cov.level` parameter.

Alternatively, a histogram can be plotted which visalizes the total number of CpGs which have been covered at different level:

```
> MEDIPS.plotSeqCoverage(seqCoverageObj = cr, type = "hist", t = 15,
+     main = "Sequence pattern coverage, histogram")
```



The parameter `t` specifies the maximal coverage depth to be plotted (here: 15 reads per sequence pattern).

5.3 CpG Enrichment

As a quality check for the enrichment of CpG rich DNA fragments obtained by the immunoprecipitation step of a MeDIP/MBD experiment, MEDIPS provides the functionality to calculate CpG enrichment values. The main idea is to test the enrichment of CpGs within the genomic regions covered by the given set of short reads compared to the full reference genome. For this, MEDIPS counts the number of Cs, the number of Gs, the number CpGs, and the total number of bases within the specified reference genome. Subsequently, MEDIPS calculates the relative frequency of CpGs and the observed/expected ratio of CpGs present in the reference genome. Additionally, MEDIPS calculates the same for the DNA sequences underlying the given regions. The final enrichment values result by dividing the relative frequency of CpGs (or the observed/expected value, respectively) of the regions by the relative frequency of CpGs (or the observed/expected value, respectively) of the reference genome.

```
> er = MEDIPS.CpGenrich(file = bam.file.hESCs.Rep1.MeDIP, BSgenome = BSgenome,
+     chr.select = chr.select, extend = extend, shift = shift,
+     uniq = uniq)
```

The enrichment results object (here `er`) contains several slots which show the number of Cs, Gs, and CpGs within the reference genome and within the given regions. Additionally, there are slots that show the relative frequency as well as the observed/expected CpG ratio within the reference genome and

within the given regions. Finally, the slots `enrichment.score.relH` and `enrichment.score.GoGe` indicate the enrichment of CpGs within the given regions compared to the reference genome. For short reads derived from Input experiments (i.e. sequencing of none-enriched DNA fragments), the enrichment values should be close to 1. In contrast, a MeDIP-seq experiment should return CpG rich sequences what will be indicated by increased CpG enrichment scores.

6 Case study: Genome wide methylation and differential coverage between two conditions

Here, we calculate genome wide short read coverage and methylation profiles for the three hESCs and for the three DE replicates [Chavez et al., 2010]. In addition, we calculate differential coverage (here this isinterpreted as differential methylation) between both conditions.

6.1 Data Import and Preprocessing

First, we create a MEDIPS SET for the first replicate from the undifferentiated hESCs:

```
> hESCs_MeDIP = MEDIPS.createSet(file = bam.file.hESCs.Rep1.MeDIP,
+     BSgenome = BSgenome, extend = extend, shift = shift, uniq = uniq,
+     window_size = ws, chr.select = chr.select)
```

where the parameters remain as defined in section `Preparations`. Subsequently, we have to concatenate the remaining two replicates to the first MEDIPS SET resulting in a list of MEDIPS SETs. In principle, this would look like:

```
> bam.file.hESCs.Rep2.MeDIP = system.file("extdata", "hESCs.MeDIP.Rep2.chr22.bam",
+     package = "MEDIPSData")
> hESCs_MeDIP = c(hESCs_MeDIP, MEDIPS.createSet(file = bam.file.hESCs.Rep2.MeDIP,
+     BSgenome = BSgenome, extend = extend, shift = shift, uniq = uniq,
+     window_size = ws, chr.select = chr.select))
```

However, here we load the preprocessed lists of MeDIP-seq MEDIPS SETs available in the `MEDIPSData` package:

```
> data(hESCs_MeDIP)
> data(DE_MeDIP)
```

For the Input-seq data sets, we explicitely create MEDIPS SETs (also named INPUT SETs) as follows

```
> hESCs_Input = MEDIPS.createSet(file = bam.file.hESCs.Input, BSgenome = BSgenome,
+     extend = extend, shift = shift, uniq = uniq, window_size = ws,
+     chr.select = chr.select)
> DE_Input = MEDIPS.createSet(file = bam.file.DE.Input, BSgenome = BSgenome,
+     extend = extend, shift = shift, uniq = uniq, window_size = ws,
+     chr.select = chr.select)
```

In principle, it is possible to add as many samples to each list as available per condition. This is valid for both the MeDIP and Input sets.

For CpG density dependent normalization of MeDIP-seq data, we need to generate a coupling set. The coupling set must be created based on the same reference genome, the same set of chromosomes, and with the same window size used for the MEDIPS SETs. For this, we specify the first MEDIPS SET in the `hESCs` object as reference, but any of the other MEDIPS SETs would be fine as well, because all of them consist of the same set of chromosomes (here only chr22, hg19) and have been generated with the same window size:

```
> CS = MEDIPS.couplingVector(pattern = "CG", refObj = hESCs_MeDIP[[1]])
```

6.2 Coverage, methylation profiles and differential coverage

In principle, it is possible to calculate genome wide coverage and methylation profiles for only one MEDIPS SET or for only one group of MEDIPS SETs using the function `MEDIPS.meth`. However, in this case study we also want to calculate differential methylation between two conditions. As soon as two groups of MEDIPS SETs are provided to the `MEDIPS.meth` function differential coverage will be calculated.

```
> mr.edgeR = MEDIPS.meth(MSet1 = hESCs_MeDIP, MSet2 = DE_MeDIP,
+     CSet = CS, ISet1 = hESCs_Input, ISet2 = DE_Input, p.adj = "bonferroni",
+     diff.method = "edgeR", prob.method = "poisson", MeDIP = T,
+     CNV = F, type = "rpkm", minRowSum = 1)
```

Parameters that can be specified are:

- `MSet1`: first group of MEDIPS SETs. Please specify at least one set.
- `MSet2`: second group of MEDIPS SETs. Differential coverage will be calculated, if MSet1 and MSet2 are not empty.
- `CSet`: a coupling set (typically a CpG coupling set). Mandatory for normalization of MeDIP data. Can be skipped, if MeDIP=F (see below).
- `ISet1`: first group of INPUT SETs (corresponding to MSet1).
- `ISet2`: second group of INPUT SETs (corresponding to MSet2). Copy number variations will be calculated, if ISet1 and ISet2 are not empty. Can be swithced off by setting CNV=F (see below).
- `p.adj`: in order to correct p.values derived from the differential coverage analysis for multiple testing, MEDIPS uses Rs' `p.adjust` function. Therefore, the following methods are available: holm, hochberg, hommel, bonferroni, BH, BY, fdr, none.
- `diff.method`: method for calculating differential coverage. Available methods: `ttest` and `edgeR`. The `ttest` method will be calculated only in case there are at least three replicates/MEDIPS SETs per group. The `ttest` method can be applied to the rpkm or CpG density normalized rms values as specified by the `type` parameter (see below). The `ttest` method

adds four vectors to the result table: `score.log2.ratio`, `score.p.value`, `score.adj.p.value`, and `score` where `score = (-log10(p.value)*10)*log(ratio)`. As an alternative, `edgeR` can be applied for testing differential coverage in genome wide windows even for less than 3 replicates per group. However, in case there is only one MEDIPS SET per group, the dispersion will be set to 0.01 (please consider section "What to do if you have no replicates" of edgeR's User's Guide). The weighted trimmed mean of M-values (TMM) method is used to calculate scale factors between libraries and the library whose upper quartile is closest to the mean upper quartile is used. The `edgeR` method will be applied to the counts of the genome wide windows. The `edgeR` method adds four vectors to the result table which are edgeR's exactTest standard output: `edgeR.logFC`, `edgeR.logCPM`, `edgeR.p.value`, and `edgeR.adj.p.value`.

- `prob.method`: Provided that the parameter `MeDIP` is set to TRUE, MEDIPS will calculate CpG density dependent probability values in order to estimate the methylation status of genome wide windows. For this, MEDIPS calculates a series of CpG coupling factor dependent probability distributions. The methylation status of each window will be estimated by its according distribution. There are two probability distributions available: `poisson` and `negBinomial`. Please note that we consider this method as poorly conceived and under further development.

- `MeDIP`: This parameter determines, if a CpG density dependent normalization values (rms and prob) will be calculated for the MEDIPS SETs given at the slots `MSet1` and `MSet2`.

- `CNV`: In case there are INPUT SETs provided at both Input slots (i.e. `ISet1` and `ISet2`), copy number variation will be tested by applying the package `DNAcopy` to the window-wise log-ratios calculated based on the the means per group. By setting `CNV`=F this function will be disabled. Please note, there is the function `MEDIPS.addCNV` which allows to run the CNV analysis on two groups of INPUT SETs using another (typically increased) window size. Subsequently, the `MEDIPS.addCNV` function matches its CNV results to a given result table as previously generated by the `MEDIPS.meth` function. This allows for calculating CNVs for larger windows what might be more appropriate in case of low sequencing depth of Input-seq data. Moreover, the time consuming CNV calculation is accelerated for larger windows and the results will be matched to a `MEDIPS.meth` derived result table previously generated for smaller windows.

- `type`: In case `diff.method` has been set to `ttest`, this parameter specifies, if differential coverage is calculated based on the rpkm or rms values. This parameter is ignored in case the edgeR method is selected as the underlying model requires count data.

- `chr`: specify one or several chromosomes (e.g. c("chr1", "chr2")), if only a subset of available chromosomes have to be processed.

- `minRowSum`: number of reads per window, that are required for the test for differential methylation (default=1).

The returned object (here `mr.edgeR`) is a list of vectors (all of the same length) where the rows correspond to genome wide windows and will be considered as the result table. The number of vectors depends on the number of provided MEDIPS SETs, the number of INPUT SETs, and on the specification of several parameters including `diff.methods`, `MeDIP`, and `CNV`. For each window and for each MEDIPS or INPUT SET, there will be vectors for the `counts` and `rpkm` values. For MEDIPS SETs there can be vectors for `rms` and `prob` values (if `MeDIP`=T). For each group with more than one provided MEDIPS or INPUT SETs, respectively, there will be an additional vector containing the means over the counts and rpkm values (and for rms and prob, if available). Moreover, there are vectors for log2 ratios (MSet1/MSet2), p.values, and adjusted p.values when differential coverage has been calculated. Optionally, there will be a vector of log2 ratios (ISet1/ISet2) as a result of the CNV analysis, in case two groups of INPUT SETs have been provided (and in case `CNV`=T) or in case a CNV analysis has been added afterwards by applying the function `MEDIPS.addCNV` (see below).

6.3 Differential coverage: select significant windows

As we have processed two groups of MEDIPS SETs, we are now interested in genomic windows which show significant differential coverage:

```
> mr.edgeR.s = MEDIPS.selectSig(results = mr.edgeR, p.value = 0.1,
+     adj = T, ratio = NULL, bg.counts = NULL, CNV = F)

Total number of windows: 513046
Number of windows tested for differential methylation: 220995
Remaining number of windows with adjusted p.value<=0.1: 51
```

Here, we set a thresold of 0.1 for the adjusted p.values in order to select significant windows. There are the following filter criteria available:

- `p.value`: this is the p.value threshold as calculated either by the `ttest` or `edgeR` method
- `adj`: this parameter specifies whether the p.value or the adjusted p.values is considered
- `ratio`: this parameter sets an additional thresold for the ratio where the ratio is either `score.log2.ratio` or `edgeR.logFC` depending on the previously selected method. Please note, the specified value will be transformed into log2 internally.
- `bg.counts`: as an additional filter parameter, it is possible to require a minimal number of reads per window in at least one of the MEDIPS SET groups. For this, the mean of counts per group is considered. The parameter `bg.counts` can either be a concrete integer or an appropriate column name of the result matrix. By specifying a column name (here e.g. `bg.counts="hESC.Input.bam.counts"` or `bg.counts="MSets1.counts.mean"`), the 0.95 quantile of the according genome wide count distribution is determined and used as a minimal background threshold (please note, only count columns are reasonable).

- `CNV`: The information on CNVs present in the samples of interest can be used for correcting differential coverage observed in the corresponding IP data (e.g. MeDIP or ChIP data). In the given example data, we do not expect extensive events of copy number variation between pluripotent and differentiated hESCs because both samples have been derived from the same cell line. However, samples derived from e.g. cancer tissues might show an relevant occurence of copy number variations. Therefore, when comparing MeDIP- or ChIP-seq data derived from cancer tissue against MeDIP- or ChIP-seq data derived from corresponding healthy tissue, an event of differential methylation might be explained in part or entirely by a local CNV. In case Input data has been provided for both conditions, MEDIPS is capable of calculating genome wide CNV ratios by employing the package DNAcopy. In case the parameter `CNV` is set to TRUE, MEDIPS will consider only genomic windows having a CNV corrected IP ratio higher than the specified ratio threshold (specification of the ratio parameter is required in this case).

6.4 Merge frames

After having identified and extracted genomic windows showing significant differential coverage between conditions, it remains of interest to merge neighboring significant windows into a larger continuous region. Please note, the result object `mr.edgeR.s` returned by the `MEDIPS.selectSig` contains genomic regions that show differential coverage into both directions, i.e. either a higher coverage in the group of samples in MSet1 (here hESCs) over the group of samples in MSet2 (here DE) or vice versa. In order to isolate the subset of genomic regions that show e.g. higher coverage in MSet2 over MSet1, we use the following R syntax:

```
> mr.edgeR.s.gain = mr.edgeR.s[which(mr.edgeR.s[, grep("logFC",
+     colnames(mr.edgeR.s))] < 0), ]
```

Subsequently, the function `MEDIPS.mergeFrames` can be applied to merge all adjacent significant regions into one region which will be finally regarded as one event of differential coverage:

```
> mr.edgeR.s.gain.m = MEDIPS.mergeFrames(frames = mr.edgeR.s.gain,
+     distance = 0)
```

Here we interpret the selected and merged genomic regions as differentially methylated regions (DMRs) showing gain of methylation during differentiation.

The `distance` parameter allows to merge windows having an according gap in between. Please note, merged windows are represented only by their genomic coordinates and do not contain any summarized values.

6.5 Regions of interest

MEDIPS provides the functionality to select subsets of the result matrix returned by the `MEDIPS.meth` function according to any given set of regions of interest (ROIs). As an example, we consider the set of merged windows (here `mr.edgeR.s.loss.m`, see previous section) as a set of ROIs:

```
> rois = MEDIPS.selectROIs(results = mr.edgeR, rois = mr.edgeR.s.loss.m,
+     columns = "counts", summarize = F)
```

The function `MEDIPS.selectROIs` will select all genomic windows in the original result table (here `mr.edgeR`) which are included in the boundaries defined by the genomic coordinates of the given set of ROIs. The number of returned windows depends on the window size and on the length of the provided ROIs. Please note, only selected columns will be returned as determined by the `columns` parameter. In this example, the function selects all columns which contain count values. In addition, it is possible to specify concrete column names and to provide several identifiers of interest like e.g. c("counts", "rpkm").

As an alternative for extracting all windows included in the genomic ranges of the given ROIs, it is possible to calculate mean values over the individual windows for each ROI, a behaviour that is controlled by the parameter `summarize`.

```
> rois.s = MEDIPS.selectROIs(results = mr.edgeR, rois = mr.edgeR.s.m,
+     columns = "counts", summarize = T)
```

For each ROI and for each specified column, there will be one mean value returned.

7 Miscellaneous

In this section we discuss several additional functionalities currently available in the MEDIPS package.

7.1 Calibration Plot

MeDIP-seq data is transformed into genome wide relative methylation scores by a previously presented CpG dependent normalization method [Chavez et al., 2010]. Normalization is based on the dependency between short read coverage and CpG density at genome wide windows. This dependency has been originally demonstrated by Down et al. [Down et al., 2008] and can be visualized as a calibration plot.

```
> MEDIPS.plotCalibrationPlot(CSet = CS, main = "Calibration Plot",
+     MSet = hESCs_MeDIP[[1]], plot_chr = "chr22", rpkm = TRUE,
+     xrange = TRUE)
```

- `CS`: the COUPLING SET
- `ISet`: an INPUT SET (i.e. a MEDIPS SET created from Input sequening data). No linear regression will be calculated for the Input Set.
- `main`: the main of the figure
- `MSet`: the MEDIPS SET object
- `plot_chr`: the chromsome for which the calibration plot will be generated

- `rpkm`: specifies, if the MeDIP data range will be plotted in count or rpkm format. It is necessary to set rpkm=T, if both, a MSet and an ISet are given and plotted at the same time.
- `xrange`: if set to TRUE, only the lower data range of the calibration plot will be visualized

Please note, it is strongly recommended to direct the output to a graphics device like e.g. png().

7.2 Export Wiggle Files

MEDIPS allows to export genome wide coverage profiles as wiggle files for visualization in common genome browsers.

```
> MEDIPS.exportWIG(Set = hESCs_MeDIP[[1]], file = "hESC.MeDIP.rep1.wig",
+     format = "rpkm", descr = "")
```

- `Set`: a MEDIPS or COUPLING SET. In case of a COUPLING SET, the `format` parameter must be set to `pdensity` because in this case a sequence pattern (e.g. CpG) density profile will be exported.
- `file`: the output file name
- `format`: can be either count or rpkm for a MEDIPS SET or pdensity for a COUPLING SET.
- `descr`: a track description for the wiggle file

7.3 Merge MEDIPS SETs

A MEDIPS SET represents the mapping results of a biological sample of interest, typically a biological or technical replicate. However, a biological or technical replicate might be sequenced several times or distributed over different lanes resulting in several bam files. Instead of merging associated bam files previously to a MEDIPS analysis, the MEDIPS package allows to merge MEDIPS SETs generated from associated bam files.

```
> Input.merged = MEDIPS.mergeSets(MSet1 = hESCs_Input, MSet2 = DE_Input,
+     name = "Input.hESCs.DE")
```

Please note, MEDIPS SETs to be merged are required to be previously generated based on the same reference genome, chromosome set, and window size. The `extend`, `shift` or `uniq` parameters can be different and are left blank in the resulting MEDIPS SET specification. The parameter `name` assigns a new file name to the merged MEDIPS SET. Consequently, the merged MEDIPS SET does not have a concrete path and file association anymore and cannot be used as input for the `MEDIPS.addCNV` function.

7.4 Correlation of MEDIPS SETs

Genome wide short read coverage profiles are expected to be similar for biological or technical replicates assuming a sufficient sequencing depth (compare section `Saturation Analysis`). The `MEDIPS.correlation` function comapares genome wide coverage profiles of provided MEDIPS SETs and returns a correlation matrix containing pair-wise Pearson correlations.

```
> cor.matrix = MEDIPS.correlation(MSets = c(hESCs_MeDIP, DE_MeDIP,
+     hESCs_Input, DE_Input))
```

7.5 Annotation of a (filtered) result table

Typically, it is of interest to annotate genomic windows with known genome characteristics like transcription start sites, exons, CpG islands etc. MEDIPS provides a basic function for adding additional annotation columns to the result table. For this, an annotation object is required which contains the annotation ids, chromosome names, start and end positions. As an example, MEDIPS allows to generate such an annotation object by accessing BioMart [Durinck et al., 2009]:

```
> anno.mart.gene = MEDIPS.getAnnotation(dataset = c("hsapiens_gene_ensembl"),
+     annotation = c("GENE"), chr = "chr22")
```

Annotation of the result matrix (either the full matrix or a subset of the matrix as selected by e.g. the `MEDIPS.selectSig` function) can now be performed as follows:

```
> mr.edgeR.s = MEDIPS.setAnnotation(regions = mr.edgeR.s, annotation = anno.mart.gene)
```

An annotation is assigned to a genomic window, if their genomic coordinates overlap.

7.6 addCNV

In case there are INPUT SETs provided at both Input slots (i.e. `ISet1` and `ISet2`) of the `MEDIPS.meth` function, copy number variation will be tested by applying the package `DNAcopy` to window-wise log-ratios calculated based on the the means per group. However, by setting `CNV`=F copy number variation analysis will be disabled avoiding a computational demanding CNV analysis at previously determined (typically short) genome wide windows. The function `MEDIPS.addCNV` allows to run the CNV analysis on two groups of INPUT SETs using another (typically increased) window size. Subsequently, the `MEDIPS.addCNV` function matches its CNV results to the given result table as previously generated by the `MEDIPS.meth` function. This allows for calculating CNVs for larger windows what might be more appropriate in case of low sequencing depth of Input-seq data.

```
> mr.edgeR = MEDIPS.addCNV(cnv.Frame = 10000, ISet1 = hESCs_Input,
+     ISet2 = DE_Input, results = mr.edgeR)
```

7.7 Comments on the experimental design and Input data

There are two common questions for conducting and analyzing MeDIP or any other kind of enrichment/immunoprecipitation (IP) based sequencing data: first, it is of interest to identify enriched regions in an IP sample in order to reveal e.g. methylated regions, the occurrence of histone modifications, or to localize transcription factor binding sites. Second, it is of interest to identify differential short read coverage comparing two conditions (here treatment vs. control) in order to identify genomic regions that show e.g. differential methylation or differential transcription factor binding. An intutive approach for identifying differential coverage is to first calculate condition-wise enrichments (or 'peaks') followed by mutual subtraction of these 'peak sets'. However, this approach has several limitations (not discussed here) and the MEDIPS package follows an alternative strategy directly comparing the groups of IP samples against each other.

For the identification of sample-wise local enrichments, any peak finder appropriate for the analyzed data type may be applied (differences of peak finding tools are not discussed here). In addition, it is recommended to include Input data for estimating a background coverage distribution and local coverage bias. Most peak finders are capable of incorporating Input data. For MeDIP data, a CpG density bias has been observed that is not considered by standard peak callers. MEDIPS allows to incorporate the CpG density into the short-read counts resulting in relative methylation values (rms), and we have shown an improved correlation of rms values to bisulfite sequencing data compared to counts or rpkm values [Chavez et al., 2010]. However, the resulting range of rms values depends on several unknown factors including the total abundance of methyalted cytosines in the sample or high outlier signals which might be present even in Input data. In case reference bisulfite data of selected genomic regions is available (low to high methylation), it is possible to associate the range of MeDIP-seq derived rms values to the bisulfite derived data range (i.e. percentage of methylated cytosines per window) [Grimm et al., 2013]. However, without reference data, it remains an insufficiently solved problem to estimate thresholds for low, intermediate, and high metylation, or to caluculate significant local enrichments based on CpG density corrected data in order to identify methylated regions (or 'peaks') over background.

For the identification of differential coverage, it can be assumed that any experiment independent bias, like local CpG density, affects the short read coverage in both samples (control and treatment) the same way. Therefore, no normalization of CpG density or other experiment independent factors may be performed when differential coverage between two groups of IP samples is calculated. As a consequence, Input samples for the control and for the treatment samples might not be necessary. However, there can exist differences in the genomic backgrounds of the control and treatment samples caused by e.g. copy number variations in cancer versus healthy cells. In such cases, it is necessary to explicitly create independent Input samples for the control and for the treatment samples in order to reveal local background differences that can influence the identification of differential coverage comparing the according IP samples. MEDIPS (1.10.0 or higher) can process an arbitrary number of Input samples per condition, if available, and allows for considering copy number variations observed in the Input data when differential coverage is calculated for the IP

data.

In summary, the need for Input data depends on two conditions: the experimental design ("Am I interested in IP enrichments over Input or do I want to calculate differential coverage between groups of IP samples?") and the genomic background or different experimental conditions of the biological samples ("Do my samples have the same genomic background?").

References

Lukas Chavez, Justyna Jozefczuk, Christina Grimm, Joern Dietrich, Bernd Timmermann, Hans Lehrach, Ralf Herwig, and James Adjaye. Computational analysis of genome-wide dna methylation during the differentiation of human embryonic stem cells along the endodermal lineage. *Genome Res*, Aug 2010.

Thomas A Down, Vardhman K Rakyan, Daniel J Turner, Paul Flicek, Heng Li, Eugene Kulesha, Stefan Graef, Nathan Johnson, Javier Herrero, Eleni M Tomazou, Natalie P Thorne, Liselotte Baeckdahl, Marlis Herberth, Kevin L Howe, David K Jackson, Marcos M Miretti, John C Marioni, Ewan Birney, Tim J P Hubbard, Richard Durbin, Simon Tavare, and Stephan Beck. A bayesian deconvolution strategy for immunoprecipitation-based dna methylome analysis. *Nat Biotechnol*, 26(7):779–785, Jul 2008.

Steffen Durinck, Paul T. Spellman, Ewan Birney, and Wolfgang Huber. Mapping identifiers for the integration of genomic datasets with the r/bioconductor package biomart. *Nature Protocols*, 2009.

Christina Grimm, Lukas Chavez, Mireia Vilardell, Alexandra Farrall, Sascha Tierling, Julia Boehm, Phillip Grote, Matthias Lienhard, Joern Dietrich, Bernd Timmermann, Joern Walter, Michal Schweiger, Hans Lehrach, Ralf Herwig, Bernhard Herrmann, and Markus Morkel. Dna-methylome analysis of mouse intestinal adenoma identifies a tumour-specific signature that is partly conserved in human colon cancer. *PLoS Genetics*, Feb 2013.

Ben Langmead, Cole Trapnell, Mihai Pop, and Steven Salzberg. Ultrafast and memory-efficient alignment of short dna sequences to the human genome. *Genome Biology*, Mar 2009.

Robinson MD, McCarthy DJ, and Smyth GK. edger: a bioconductor package for differential expression analysis of digital gene expression data. *Bioinformatics*, Jan 2010.

Venkatraman E. Seshan and Adam Olshen. Dnacopy: Dna copy number data analysis. *Bioconductor*.

Michael Weber, Jonathan J Davies, David Wittig, Edward J Oakeley, Michael Haase, Wan L Lam, and Dirk Schuebeler. Chromosome-wide and promoter-specific analyses identify sites of differential dna methylation in normal and transformed human cells. *Nat Genet*, 37(8):853–862, Aug 2005.