Chapter 10

Quantitative Comparison of Large-Scale DNA Enrichment Sequencing Data

Matthias Lienhard and Lukas Chavez

Abstract

DNA enrichment followed by sequencing (DNA-IP seq) is a versatile tool in molecular biology with a wide variety of applications. Computational analysis of differential DNA enrichment between conditions is important for identifying epigenetic alterations in disease compared to healthy controls and for revealing dynamic epigenetic modifications throughout normal and distorted cell differentiation and development. We present a protocol for genome-wide comparative analysis of DNA-IP sequencing data to identify statistically significant differential sequencing coverage between two conditions by considering variation across replicates. The protocol provides a detailed description for the comparative analysis of DNA-IP sequencing data including basic data processing, quality controls, and identification of differential enrichment using the Bioconductor package "MEDIPS".

Key words ChIP-seq, DNA methylation, Sequencing, MeDIP-seq, Epigenetics, DNA enrichment, Computational biology, Bioconductor, Quality control

1 Introduction

DNA enrichment methods are widely used for genome-wide identification of many different kinds of epigenetic marks. These techniques include chromatin-immunoprecipitation for localizing transcription factor binding sites or for revealing the genomic distributions of diverse types of histone modifications. To profile chromatin dynamics of scarce in vivo cell populations, an indexing-first chromatin IP approach (iChIP) has recently been developed [1]. *Me*thylated *D*NA *I*mmuno-*P*recipitation (MeDIP) [2] and methyl-CpG binding domain (MBD) protein capture [3] are similar techniques, but target the enrichment of DNA fragments containing methylated cytosines. Similarly, 5-hydroxymethylcytosines can be detected by antiserum specific to cytosine-5-hydroxymethylenesulfate (CMS) after treatment with sodium bisulfite [4]. To identify genomic loci enriched for DNA fragments in the DNA-IP sample compared to a negative control (e.g., Input-DNA), many different

Ewy Mathé and Sean Davis (eds.), *Statistical Genomics: Methods and Protocols*, Methods in Molecular Biology, vol. 1418, DOI 10.1007/978-1-4939-3578-9_10,

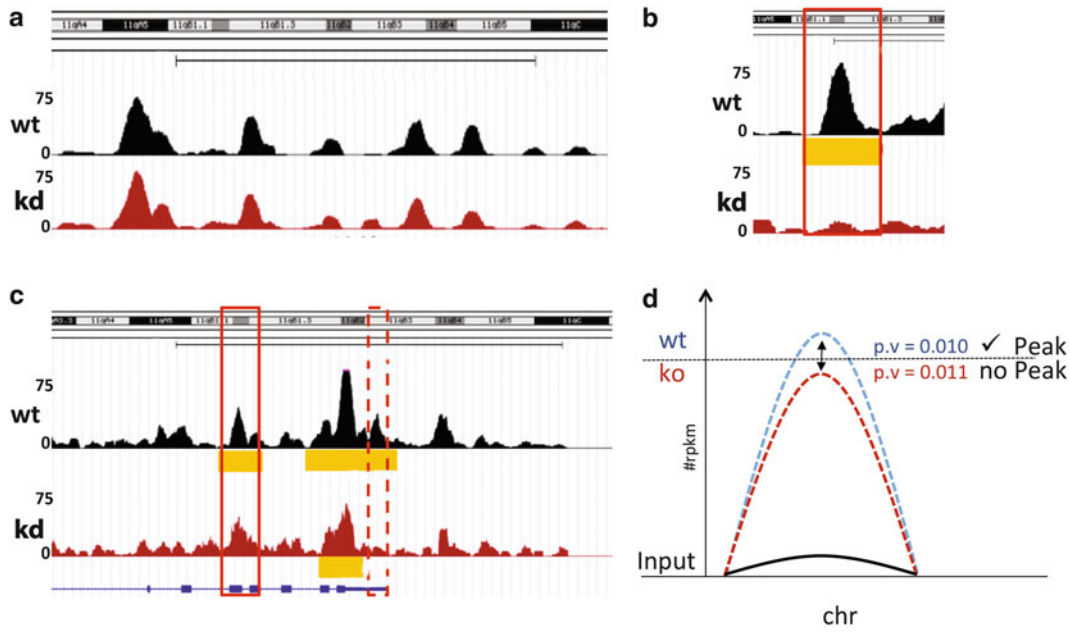

Fig. 1 Critical assessment of methods that solely rely on peaks for identifying differential enrichment between conditions. (**a**) Reproducible 5hmC enrichment in v6.5 wild type (*wt*) and Tet2 knock down (*kd*) mouse embryonic stem cells [12]. (**b**) In an ideal situation, differential enrichment is defined by the presence of a peak in one condition and the absence of enrichment in the other condition. (**c**) A peak identified in the *wt* condition has not been detected in the *kd* condition although the enrichment appears not to be depleted (*red box*). A peak detected in *wt* extends over a region with a highly variable enrichment profile throughout. Depletion of DNA enrichment in *kd* occurs in only a fraction of the peak identified in *wt* (*dashed red box*). (**d**) Schematic representation of the peak calling threshold problem that causes false positive differential enrichment. To define differential DNA-IP enrichment between conditions, MEDIPS evaluates the difference of DNA enrichment strength at distinct genomic loci instead of considering the presence or absence of previously calculated peaks ("peak free approach")

tools are available that can be categorized into the group of peak-callers [5, 6]. While these tools are capable of identifying DNA enrichments per sample, it is a common pitfall to rely on the identified sets of peaks for concluding sample specific and differential DNA enrichment comparing conditions (Fig. 1c–d). This procedure can cause false results due to several reasons. First, identification of peaks depends on statistically derived thresholds for elevated local sequencing coverage over background. This can lead to situations where a peak has been detected in one sample but not in the other although the sequencing coverage differs only slightly between samples (Fig. 1c–d). Second, peaks can be detected in both conditions without considering that there exist strong imbalances of DNA enrichment towards increased enrichment in one condition compared to the other.

Third, peaks often extend over long DNA regions incorporating several sites of epigenetic marks with distinct dynamics.

Figure 1c shows an example where the genomic coordinates of the peak identified in a control (*wt*) condition partly overlaps with a peak identified in a treatment condition (*kd*), indicating epigenetic alterations in only a fraction of the peak region. Appropriate segregation of such peaks into small fractions and subsequent differential enrichment analysis of each sub-region is mandatory to identify crucial epigenetic alterations between conditions. Finally, peak callers do not consider information on biological or technical variance across replicates. We therefore propose a method that calculates DNA-IP sequencing coverage separately for all samples at small genome wide windows [7]. Modeling the window coverage using negative binomial distribution allows for incorporation of biological variation between replicates. Based on this model, we apply a statistical test [8] to identify differentially enriched genomic regions between conditions. Afterwards, neighboring genomic regions with significant differential coverage into the same direction (i.e., either loss or gain of DNA enrichment) can be merged to define distinct loci of epigenetic alterations.

In this protocol we describe the analysis of large-scale ChIP-seq data obtained by profiling the histone modification H3K4me2 in naive and T_H2 memory (or CCR4 positive, respectively) T cells [9]. We demonstrate how to process and compare data of approx. forty distinct ChIP-seq assays per condition to ultimately identify cell type-specific enhancers marked by statistically significant differential H3K4me2 enrichment.

2 Materials

The MEDIPS analysis pipeline is implemented as an *R/Bioconductor* package, and therefore, runs on a wide variety of UNIX platforms, MS Windows and MacOS. We recommend installing the latest version of R freely available at www.r-project.org. The hardware requirement for the analysis depends on the genome size, the targeted resolution (window size), and the number of samples. We observe that processing of 84 human samples at distinct genomic windows of length 500bp requires approx. 40 GB of memory.

2.1 Installation of MEDIPS

To install the latest version of MEDIPS, it is recommended to first download and install the latest R version available at www.r-project.org. Consequently, the latest version of the MEDIPS package will be installed when employing the Bioconductor installer *biocLite*. To install MEDIPS start R and write:

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

For this protocol we have employed R version 3.1.2, which is linked to Bioconductor version 3.0 (www.bioconductor.org)

containing MEDIPS version 1.17.2. To start a MEDIPS workflow, the MEDIPS library needs to be loaded into the R environment by writing:

```
R> library(MEDIPS)
```

2.2 Preprocessing of the Sequencing Data

The typical outputs of IP-sequencing experiments are short sequencing reads together with quality scores in *fastq* format. The H3K4me2-ChIPSeq data processed in this protocol is color-space sequencing data produced by SOLiD sequencing and the raw data (*csfastq* and *qual* files) but also *fastq* files are available at GeneExpressionOmnibus under accession id GSE53646 (http://www.ncbi.nlm.nih.gov/geo/query/acc.cgi?acc=GSE53646). To reproduce the analysis steps described here, ChIP-seq data for naïve and T_H2 samples have to be downloaded (these are the samples GSM1297924 to GSM1298001). Subheading 3.6 shows an R script that accomplishes this task by employing the Bioconductor package *SRAdb*. In order to apply the MEDIPS analysis, the reads first have to be aligned to a reference genome using an alignment tool, such as bowtie [10]. For the example data, the following alignment parameters have been applied:

```
> bowtie --sam -m 1 -C hg19 sample_A_runX.fastq sample_A_runX.sam
```

The option "–sam" determines the output format, -m 1 ensures that only unique hits in the genome are reported, and -C specifies that the reads are SOLiD colorspace encoded. The script for aligning the sequencing data is given in Subheading 3.7. Each sample has been multiplexed and sequenced across several sequencing lanes and runs. Therefore, data across different lanes and runs are merged to a single sam file per sample using *samtools* (http://samtools.sourceforge.net/). In order to save memory and to facilitate fast random access to the alignments, we recommend to sort the sam files by genomic position, to convert the text based sam files into a binary and compressed bam format and to create according index files using *samtools*:

```
> samtools view -Sb sample_A_runX.sam | samtools sort - sample_A_runX
> samtools merge sample_A_runX.bam sample_A_runY.bam [...] sample_A
> samtools index sample_A.bam
```

3 Methods

In this section, we describe a MEDIPS workflow for quality control and quantitative comparison of ChIP-seq data and explain all relevant options and parameters of the given examples. The according R script that reproduces the H3K4me2 ChIP-seq data analysis of naïve and T_H2 T-cell subtypes [9] is given in Subheading 3.6. First,

we import the table containing all sample information into R (this table is generated in Subheading 3.6):

```
> samples=read.table("Seumois_NatImmu2014_sample_table.txt",
header=T, sep="\t", stringsAsFactors=FALSE)
> head(samples)
  accession        name
1 GSM1297951 CCR4Neg-Donor19-rep1
2 GSM1297950 CCR4Neg-Donor18-rep2
3 GSM1297957 CCR4Neg-Donor22-rep2
4 GSM1297969  CCR4pos-Donor7-rep1
5 GSM1297977 CCR4pos-Donor11-rep2
6 GSM1297993 CCR4pos-Donor19-rep2
                          filename celltype
1 bam/GSM1297951_H3K4me2_ChIPSeq_CCR4Neg-Donor19-rep1.bam CCR4Neg
2 bam/GSM1297950_H3K4me2_ChIPSeq_CCR4Neg-Donor18-rep2.bam CCR4Neg
3 bam/GSM1297957_H3K4me2_ChIPSeq_CCR4Neg-Donor22-rep2.bam CCR4Neg
4 bam/GSM1297969_H3K4me2_ChIPSeq_CCR4pos-Donor7-rep1.bam CCR4pos
5 bam/GSM1297977_H3K4me2_ChIPSeq_CCR4pos-Donor11-rep2.bam CCR4pos
6 bam/GSM1297993_H3K4me2_ChIPSeq_CCR4pos-Donor19-rep2.bam CCR4pos
```

In addition we have to install, load, and specify a reference BSgenome package. Here, we install and load the human reference build hg19/GRCh37:

```
> biocLite("BSgenome.Hsapiens.UCSC.hg19")
> library("BSgenome.Hsapiens.UCSC.hg19")
> reference="BSgenome.Hsapiens.UCSC.hg19"
```

3.1 Parameter Settings and Quality Control

3.1.1 Stacked Reads

Due to PCR amplification applied during experimental processing of ChIP-seq assays, library complexity can become low. In this case the same DNA fragments are sequenced multiple times what can lead to elevated amounts of "stacked" reads. Such sequencing reads will have the same sequence and will align to the same genomic position. High abundance of stacked reads is indicative for over-amplification and for libraries constructed from low quantities of input DNA. To avoid false positive differential enrichment between conditions due to stacked reads, MEDIPS offers the option to substitute each stack by only one representative by setting the *uniq* parameter of the *MEDIPS.createSet()* function to TRUE (*uniq=T*). The fraction of reads that build such stacks should be monitored by checking the standard output of MEDIPS ("*Total number of imported short reads*" vs. "*Number of unique short reads*"), and by visualizing the bam files in appropriate genome browsers. Low complexity samples should be excluded from further analyses. Please note that the concept of stacked reads is different to the concept of unique and multiple mapping reads. Multiple mappers cannot be assigned to a unique position of the reference genome whereas unique mappers can be assigned to a

unique position of the reference genome. We typically consider only unique mappers for differential enrichment analysis although it is in principle possible to consider multiple mapping positions per read. However, both multiple and unique mappers can pile up to stacked reads and will be replaced by only one representative when setting the *uniq* parameter of the *MEDIPS.createSet()* function accordingly.

3.1.2 Extend Reads

The ChIP-seq protocol applied by Seumois et al. [9] generates and selects DNA fragments of length 100–250 base pairs (bp) for sequencing. By single-end sequencing, only one end of these DNA fragments is sequenced and the exact length of the individual fragments is unknown. Because the sequenced DNA fragments are typically longer than the actual sequencing reads, MEDIPS offers the option to extend the reads to the estimated average DNA fragment length by specifying a concrete value for the *extend* parameter of the *MEDIPS.createSet()* function. Given the DNA fragment size distribution of 100–250 bp we decided to extend all sequencing reads to 120 bases (*extend* =120).

3.1.3 Window Size

The fine resolution of DNA-IP enrichment experiments for narrowing down the exact location of the epigenetic mark, or the transcription factor binding site of interest, is restricted by the length of sonicated and immunoprecipitated DNA fragments. Consequently, an optimal window size for tiling the genome into nonoverlapping consecutive genomic regions ("*windows*") is influenced by the estimated average DNA fragment length and also by the expected distribution of the analyzed epigenetic mark across the genome (broad enrichment vs. sharp peaks). When mapping histone modifications, the maximal resolution of the ChIP-seq data is limited by the 146 bp DNA sequence that is wrapped around a nucleosome. However, the higher the targeted resolution of the results, the deeper sequencing is required. Decreasing the targeted resolution of the results might compensate lower sequencing coverage, but local effects might be overseen due to data smoothening. MEDIPS allows for controlling the targeted resolution of the results by the *window_size* parameter of the *MEDIPS.createSet()* function.

3.1.4 Coverage Saturation Analysis

An import quality control is to ensure that the sequence complexity of the libraries and depth of sequencing coverage is sufficient for subsequent data analysis. In order to assess the effect of the actual sequencing depth of a given sample as well as the impact of the *window_size* parameter on the reproducibility of the results, MEDIPS offers a coverage saturation analysis that randomly splits the given sequencing data into distinct subsets of increasing size and iteratively calculates correlations between such artificially created technical replicates. Please note that high abundance of stacked

reads can have an undesired positive impact on the Pearson correlation calculated between two data sets (or data subsets, respectively). Therefore, it is recommended to either use the rank based Spearman correlation or to remove stacked reads (*uniq* = T). As a rule of thumb, an estimated correlation of around 0.9 should be observed for a sufficient sequencing coverage. If the correlation is lesser than 0.8, either additional sequencing or a larger window sizes should be considered. The actual number of reads necessary for obtaining a sufficient sequencing coverage depends on the genome size and on the genome wide abundance of the epigenetic mark of interest or on the number of transcription factor binding sites, respectively.

The coverage saturation analysis can be applied to individual samples using the function *MEDIPS.saturation()*. As input, the function requires the alignment file of a sample, the reference genome, and concrete settings for the parameters *window_size*, *extend*, and *uniq* as described above. Typically it is sufficient to apply the saturation analysis to only one selected chromosome what avoids importing the entire alignment file. The results are stored in an R list object that can be plotted by the *MEDIPS.plotSaturation()* function. The following code compares the coverage saturation of an arbitrary sample for window sizes 150 bp and 500 bp, respectively:

```
>     sat150=MEDIPS.saturation(file=samples$filename[1],
reference,
    uniq=TRUE, extend=120, window_size=150,
    chr.select="chr22")
>     sat500=MEDIPS.saturation(file=samples$filename[1],
reference,
    uniq=TRUE, extend=120, window_size=500,
    chr.select="chr22")
> par(mfrow=c(1,2))
> MEDIPS.plotSaturation(sat150, main=
    paste(samples$name[1], "Saturation analysis","\nwin-
dow size 150"))
> MEDIPS.plotSaturation(sat500, main=
    paste(samples$name[1], "Saturation analysis","\nwin-
dow size 500"))
```

As depicted in Fig. 2, the maximal estimated saturation is 0.8 at a window size of 150 bases, and improves to 0.85 at 500 bases. Based on these results, we chose a window size of 500 bp for further analyses.

3.2 Importing Alignment Data

When having determined the required parameters for *window_size*, *uniq*, and *extend,* the alignment data can be imported into MEDIPS by applying the function *MEDIPS.createSet()* to each of the samples. For each genomic window, the function counts the number of overlapping reads regardless of the fraction of the overlap.

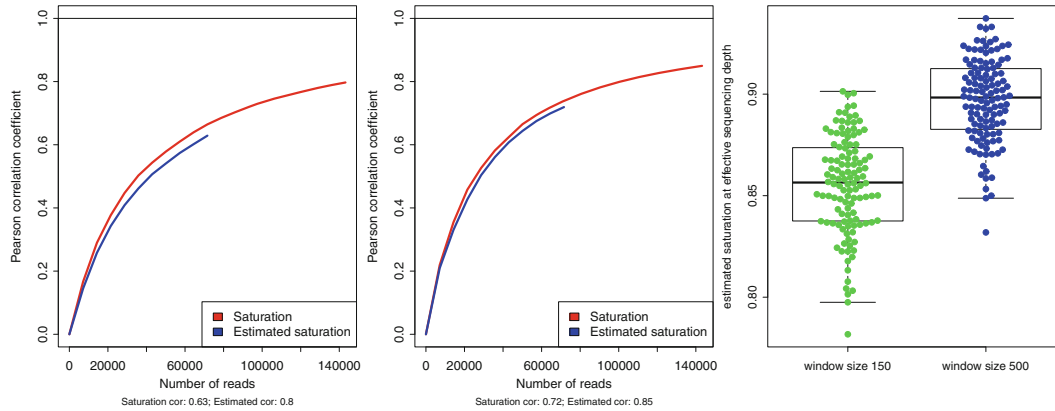

Fig. 2 Saturation analysis using window sizes of 150 bp (*left*) and 500bp (*middle*). Compared to 150 bp windows, the lower resolution of 500 bp windows leads to higher correlations (*y*-axis) between distinct random subsets throughout the tested range of sequencing depth (*x-axis*). The *x*-axis shows the increasing number of reads randomly assigned to the artificial replicates in each of the iterations. The maximally obtained estimated saturation increases from 0.8 for 150 bp windows (*left*) to 0.85 for 500 bp windows (*right*) when the entire data set is considered (maximum of the *red curve*). When applying the saturation analysis to all of the samples individually, we observe that the window size of 500 bp generally results in a higher maximal estimated saturation compared to the smaller window size of 150bp (*right, box*-and-*whisker plot*)

In addition to the three described parameters, the *MEDIPS.createSet()* function requires the alignment file (typically a bam or a bed file) and the name of the reference genome *BSgenome* package. Moreover, it is possible to restrict the analysis to a set of chromosomes by specifying the *chr.select* parameter.

```
> CCR4pos1=MEDIPS.createSet(samples$filename[1],
BSgenome=reference,
    uniq=FALSE, extend=120, window_size=500,
    chr.select=c(paste0("chr", 1:22), "chrX", "chrY") )
```

We have observed only a moderate level of stacked reads and therefore, decided to set *uniq*=FALSE for this data set (please note that such a moderate level of stacked reads is rather unusual and removal of stacked reads is typically recommended). The resulting *MSet* objects of several replicates can be concatenated into an R list, one for each experimental group. Please note, concatenating of *MSet* object does not merge the data and all replicates will be treated separately in the downstream analysis. In this study, this procedure results in two lists of *MSets*, one for naive and one for T_H2 T-cells (for the actual code please *see* Subheading 3.8). In principle, the code for three replicates per condition will look similar like this (please note that the MSets *CCR4pos2*, *CCR4pos3*, and *Naive1*, *Naive2*, *Naive3* have not been created in this example):

```
> MSet=list(
    CCR4pos=c(CCR4pos1, CCR4pos2, CCR4pos3),
    Naive=c(Naive1, Naive2, Naive3))
```

3.3 Differential Coverage Analysis

To identify genomic regions that are significantly differentially enriched between groups of samples, we will have to (1) normalize the genome wide count data for different library sizes of the samples, (2) find a minimal coverage across samples to avoid extensive unnecessary tests, (3) model variation across replicates, and (4) apply a statistical test suitable for DNA sequencing derived count data. An obvious choice for modeling the distribution of count data is the *Poisson* distribution. However, our samples are derived from different individuals with inherent biological variation among these biological replicates. This variation causes an overdispersed *Poisson* distribution, which can be described by the *negative binomial distribution*. Please note, to estimate the biological variation, replicates are required. Without replicates, biological variation will be set to a fixed value; however, this approach will always be an unsatisfactory approximation of unknown technical and biological variability. To accomplish the described tasks, MEDIPS employs the Bioconductor package *edgeR* [8] and its methods for library size normalization (*TMM*) and estimation of common and tag-wise dispersion. The function *MEDIPS.meth()* accepts two lists of *MSet* objects, each containing an arbitrary number of previously imported replicates. Moreover, the function requires further parameter settings for determining the type of statistical test applied (*diff.method*="edgeR"), the minimal number of read counts per window across replicates required for a window to be tested for differential coverage (*minRowSum*), and others (*see* also the MEDIPS vignette at http://www.bioconductor.org/packages/release/bioc/html/MEDIPS.html).

```
> resNvPos=MEDIPS.meth(MSet1=MSet[["CCR4pos"]], MSet2=MSet[["Naive"]],
diff.method="edgeR", MeDIP=F, minRowSum=10)
```

As a result, the *MEDIPS.meth()* function returns a table with test statistics for all windows. In order to depict the differences between the sets of samples, an M-vs-A-plot can be created from this table. The M-vs-A-plot contrasts the log ratios (M-value, y-axis) of the column with column name *edgeR.logFC* and the average log coverage (A-value, x-axis) of the column with column name *edgeR.logCPM* (*see* Fig. 3).

While the *MEDIPS.meth()* function returns a data table containing genome wide data, the function *MEDIPS.selectSig()* can be applied to reduce the results to those windows that fulfill the specified statistical requirements. Available parameters include a *p*-value threshold (*p.value*), the option to apply this threshold to either the raw (*adj*=*F*) or the adjusted *p*-value (*adj*=*T*), and a threshold for the coverage ratio between the two conditions (*ratio*). We term windows with significant differential DNA-IP enrichment at an adjusted *p*-value of ≤ 0.01 as differentially enriched regions (*DERs*). For visualization purpose only (*see* the

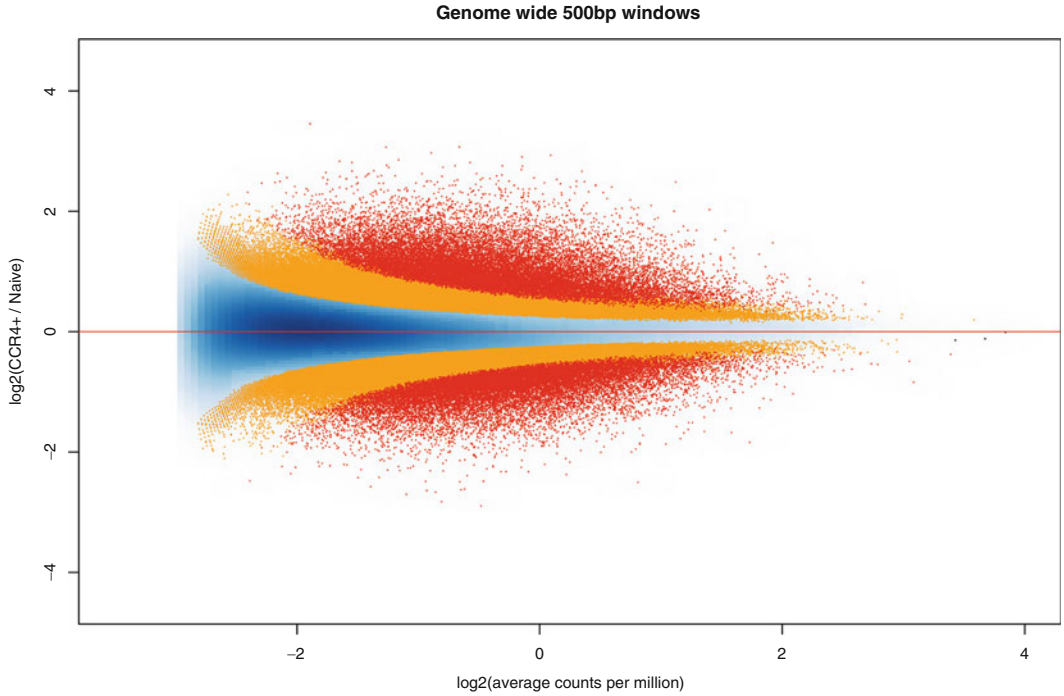

Fig. 3 "Minus-average" (MA) plots for genomic regions with differences in H3K4me2 enrichment (DERs) for comparing naïve and T_H2 T-cells. *Red* and *orange dots* indicate DERs with an adjusted $P < 0.05$ and raw $P < 0.005$, respectively. The MA plot has been generated by processing the entire ChIP-seq data set of naïve and T_H2 memory T-cells [9]

M-vs-A plot in Fig. 3), we also select windows that fulfill the relaxed threshold of a raw *p*-value ≤ 0.005*:*

```
> sigNvPos= MEDIPS.selectSig(resNvPos, p.value = 0.01,
adj = T)
> trendNvPos= MEDIPS.selectSig(resNvPos, p.value=0.005,
adj=F)
```

3.4 Merging Neighboring DERs and Annotation

DERs obtained by the *MEDIPS.selectSig()* function can be either focal events with no other immediate epigenetic differences in the vicinity, or they can be located in longer stretches of potential regulatory regions like for example super enhancers [11]. Adjacent DERs with imbalanced enrichment towards the same condition can be merged into extended and distinct loci by applying the function *MEDIPS.mergeFrames()*. Here, we are first dividing the DERs into those that gain H3K4me2 in T_H2 cells and those that lose H3K4me2 in T_H2 cells compared to naïve T-cells. Please note, log2 ratios are reported as log2(MSet1/MSet2):

```
> sigNvPosGain = sigNvPos[which(sigNvPos$edgeR.logFC>0),]
> sigNvPosLoss = sigNvPos[which(sigNvPos$edgeR.logFC<0),]
```

Afterwards, we can merge neighboring DERs:

```
sigNvPosGainMerged = MEDIPS.mergeFrames(sigNvPosGain)
sigNvPosLossMerged = MEDIPS.mergeFrames(sigNvPosLoss)
```

To further investigate potential functions of the identified DERs, MEDIPS provides functions to access the ENSEMBL database for gene annotation and to annotate the DERs by their proximity to known genes. The function *MEDIPS.getAnnotation()* connects to the ENSEMBL database and receives exon, gene or transcription start site information. The DERs can then be annotated by their proximity to the obtained annotations by applying the function *MEDIPS.setAnnotation()*. Here, we annotate merged DERs by ENSEMBL transcript names, if they are located in a promoter region (-1kb to +0.5kb around the transcription start sites):

```
> tss_ens = MEDIPS.getAnnotation(host="www.biomart.org",
dataset="hsapiens_gene_ensembl",
annotation="TSS",tssSz=c(-1000,500))
sigNvPosGainMerged = MEDIPS.setAnnotation(sigNvPosGain-
Merged, tss_ens)
sigNvPosLossMerged = MEDIPS.setAnnotation(sigNvPosLoss-
Merged, tss_ens)
```

3.5 Notes

We have described a peak-free approach for identifying statistical significant differential enrichment of DNA-IP sequencing data at distinct genome wide small regions. The described method, implemented in the Bioconductor package MEDIPS, enables the quantitative comparison of two conditions across an arbitrary number of biological replicates per group. In addition, we have demonstrated how to apply these methods for the identification of cell-type specific enhancers by comparing large scale H4K4me2 ChIP-seq data between naive and T_H2 memory T cells [9]. Although the ChIP-seq data processed in this protocol is single-end data, MEDIPS provides the functionality to process paired-end sequencing data (see the parameter *paired* of the *MEDIPS.createSet()* function).

Further functionalities of the MEDIPS package, not demonstrated in this protocol, include incorporation of Input-DNA sequencing data. In case such control data is available and genomic backgrounds vary between the samples (e.g., due to copy number variations in cancer samples compared to healthy controls), the parameter *CNV* of the function *MEDIPS.meth()* can be enabled to calculate potential CNVs based on the given Input-seq data. Subsequently, the fold change of the IP-DNA sequencing data can be corrected by the fold change of the Input-DNA sequencing data to exclude DERs that can be explained by differences in the genomic data alone. Excluding such DERs is controlled by the *ratio* and *CNV* parameters of the *MEDIPS.selectSig()* function.

The high number of statistical tests applied in such genome wide screens at small genomic windows can cause a severe multiple

testing problem in case the number of replicates is small. While the number of biological replicates available in the discussed study [9] is sufficiently high for applying stringent significance thresholds to the corrected *p*-values, experimental designs with much smaller sample numbers will likely result in almost no or only a small number of sufficiently significant adjusted *p*-values. To overcome this problem, the number of applied tests can be reduced by two different ways. First, a minimal number of reads that fall into a genomic window across all of the samples can be required prior to testing. By setting the *minRowSum* of the *MEDIPS.meth()* function, the user can specify the number of reads that must fall into a genomic window in all of the samples together, otherwise this window will not be tested for differential enrichment. We generally recommend a *minRowSum* value of at least 10 to avoid massive amounts of tests at almost non-covered genomic regions. However, this value depends on the sequencing depth of the individual samples and an optimal *minRowSum* value can be estimated by investigating the count distribution of the DNA-IP sequencing samples or of Input-DNA sequencing samples, if available. Please note, there is currently no method implemented in MEDIPS that estimates an optimal *minRowSum* parameter. Second, instead of testing genome wide genomic windows, MEDIPS can be applied to any set of predefined regions of interest. Instead of a *window_size* parameter, the function *MEDIPS.createROIset()* accepts the genomic coordinates of any set of regions of interest as input for its *ROI* parameter (see also the man page of this function available by writing *?MEDIPS.createROIset* in R).

Besides applying the statistical test implemented in edgeR [8], MEDIPS allows for deriving *p*-values by applying the t-test to each of the genomic windows, if at least three replicates are available per condition (see the *diff.method* and *type* parameters of the *MEDIPS.meth()* function).

Varying enrichment efficiencies of DNA-IP sequencing experiments can have a negative impact on proper differential enrichment analysis when comparing DNA-IP assays from different batches. We have previously shown that systematic ChIP-seq batch effects can introduce likely false positive differential enrichment between conditions [9]. To approach this issue, MEDIPS version $\geq$1.18.0 enables Quantile normalization of the read counts across all samples prior to testing for differential enrichment between conditions.

The MEDIPS package provides further functionalities for DNA-IP sequencing data processing, including export of Wiggle files for visualization of DNA-IP sequencing data in genome browsers (see *MEDIPS.exportWig()*), merging of data sets in R to avoid unnecessary merging of bam files (see *MEDIPS.mergeSets()*), and others. Moreover, MEDIPS maintains its functionalities specific for MeDIP-seq experiments, including CpG density normalization and calculation of relative methylation scores. The MeDIP-seq specific functionalities can be enabled by the *MeDIP* parameter of

the *MEDIPS.meth()* function. A complete list of functions is available in the reference manual at http://www.bioconductor.org/packages/release/bioc/html/MEDIPS.html.

3.6 R Script to Download the H3K4me2 ChIP-seq Data from SRA

```
#This script demonstrates how to download the entire human ChIP-seq data (SRA id: SRP034717) presented by Seumois et al. [9] into the designated working directory. The entire data set consists of 1789 fastq files across several multiplexed lanes and runs that can be assigned to 120 distinct ChIP-seq samples.

#Install the SRAdb package
source('http://bioconductor.org/biocLite.R')
biocLite('SRAdb')

#Load the SRAdb library and download the latest database content
library(SRAdb)
srafile = getSRAdbFile()  #download of a SRA database snapshot (~800 mb)
sra_con = dbConnect(SQLite(),srafile)

#Fetch a table with the relevant run and sample information
runs = getSRA( "SRP034717", out_types="sra", sra_con)
runs = unlist(strsplit(paste0(runs$run,": ",runs$experiment_title),"[:;] "))
runs = as.data.frame(matrix(runs,ncol=5, byrow=T))

#For this protocol we restrict the analysis to only a small subset of the samples.
#Here, we select three T_H2 and three naïve T-cell samples:
subselection=c("GSM1297960",  "GSM1297962",  "GSM1297964",  "GSM1298002",
"GSM1298005", "GSM1298007")
runs=runs[(runs[,2]%in%subselection),] #remove this line, if the entire data set should be processed

#Create a fastq table that contains the necessary information to link the individual fastq files to their respective samples (in this example there are 70 fastq files that can be assigned to the six selected sample):

write.table(runs,  "Seumois_NatImmu2014_run_table.txt",  col.names=F,  row.
names=F, sep="\t", quote=F)

#Create a sample table that lists the sample names, their GO id and the anticipated location of the bam files in a separate bam sub-folder. The bam files will be generated based on the downloaded fastq files as shown in Subheading 3.7.
samples = unique(runs[,2:3])
samples[,2] = sub("H3K4me2_ChIPSeq_","",samples[,2])
names(samples) = c("accession", "name")
samples$filename = paste0("bam/", samples$accession, "_H3K4me2_ChIPSeq_",sam-
ples$
          name,".bam")
samples$celltype=sub("-Donor.+$","",samples[,2])
```

```
write.table(samples, "Seumois_NatImmu2014_sample_table.txt",
   col.names=T, row.names=F, sep="\t", quote=F)
#Download the fastq files into a fastq sub-folder:
#Note that the reduced set of 6 samples is about 7.5 gb, the complete dataset is
                    about 152 gb
dir.create("fastq")
for(acc in samples$accession){
 sra=runs[runs[,2]==acc,1]
 getSRAfile(in_acc=sra, sra_con=sra_con, fileType="fastq", destDir="fastq")
}
```

3.7 Shell Script for the Alignment of the H3K4me2 ChIP-seq Data (Fastq Files)

```
#!/bin/bash
#Set the path to your bowtie index (here this is a hg19 color-
space index):
hg19=/path/to/bowtie/genome/reference/hg19_CS
#specify number of cores
n_cores=30

#Set the variable dir to the designated working directory which is supposed to be
the same working directory as for the R script in Subheading 3.6. The working
directory is supposed to contain the fastq sub-folder containing the downloaded
fastq files:
dir=`pwd`
#create the sub-folder that will contain the resulting alignment bam files
mkdir bam

#Assign the information of the fastq table (created in Subheading 3.6) to three
separate list variables:
run=($(cut -f1 Seumois_NatImmu2014_run_table.txt)) #Unique IDs for the 70 indi-
vidual fastq files
sampleAcc=($(cut -f2 Seumois_NatImmu2014_run_table.txt))  #GEO sample IDs
sampleName=($(cut -f3 Seumois_NatImmu2014_run_table.txt)) #Sample names

#Assuming that your PATH variable contains the path to bowtie and samtools bin-
aries and assuming that your compute server has at least 30 cores, the fastq files
can now be aligned and sorted into sample specific bam files as follows:
#The computation of the alignment takes about 30 minutes per sample on 30 cores
for s in $(printf '%s\n' "${sampleAcc[@]}"|sort|uniq)
do
list=""
skip="no"
nrruns=0
#get all fastq files for sample $s
for (( i=0; i<${#run[@]}; i++ ))
do
if [ ${sampleAcc[$i]} = $s ]; then
fq=${dir}/fastq/${run[$i]}.fastq.gz
if [ -e $fq ]; then
```

```
list="$list $fq"
nrruns=$((nrruns+1))
sName=${sampleName[$i]}
else
echo "warning: [sample $s] $fq not found"
skip="yes"
fi
fi
done
if [ $nrruns -ge 1 -a $skip != "yes" ]; then
echo "aligning reads for $s from $nrruns runs"
#
cat $list | gunzip |bowtie -sam -n 2 -m 1 -C -p 30 $hg19 - | \
   samtools view -Sb - | \
   samtools sort - ${dir}/bam/${s}_${sName}
samtools index ${dir}/bam/${s}_${sName}.bam &
fi
done

#This script also creates bam index files for each of the six resulting bam files.
```

3.8 R Script for Quality Control and Differential Enrichment Analysis of the Aligned H3K4me2 ChIP-seq Data Comparing T_H2 and Naïve T-Cells

```
#Go to the designated working directory as for Subheadings 3.6 and 3.7 and start R. This working directory is supposed to contain a bam sub-folder containing the bam files created in Subheading 3.7.

library(MEDIPS)
library(BSgenome.Hsapiens.UCSC.hg19)
samples=read.table("Seumois_NatImmu2014_sample_table.txt",
   header=T, sep="\t", stringsAsFactors=FALSE)
dir.create("plots")
#Coverage saturation analysis for all samples and for two different window sizes each:
reference="BSgenome.Hsapiens.UCSC.hg19"
saturation=data.frame(ws150=numeric(), ws500=numeric())
#The saturation analysis should take < 1 minute per sample (depending on speed of HDD)
for (i in 1:nrow(samples) ){
sat150=MEDIPS.saturation(file=samples$filename[i],reference,
     uniq=TRUE, extend=120, window_size=150,
     chr.select="chr22")
sat500=MEDIPS.saturation(file=samples$filename[i],reference,
     uniq=TRUE, extend=120, window_size=500,
     chr.select="chr22")
```

```
saturation[samples$name[i],] =
   c(sat150$maxEstCor[2],sat500$maxEstCor[2])
png(file=paste0("plots/saturation_",samples$accession[i],".png"),
   width=800, height=400 )
par(mfrow=c(1,2))
MEDIPS.plotSaturation(sat150, main=
paste(samples$name[i], "Saturation analysis","\nwindow size 150"))
MEDIPS.plotSaturation(sat500, main=
paste(samples$name[i], "Saturation analysis","\nwindow size 500"))
dev.off()
}

#Import of bam files: for each sample an MSet object is created and the MSet objects
are concatenated into a list of two separate MSet list objects. The individual
MSets can be assigned to their respective cell types by their list names. We
restrict the imported mapping data to the major chromosomes:
#reading the alignment files and computing the coverage takes about 10 minutes per
sample (depending on speed of HDD)
MSet=list()
for (i in 1:nrow(samples) ){
MSet[[samples$celltype[i]]]=c(MSet[[samples$celltype[i]]],
  MEDIPS.createSet(samples$filename[i],
BSgenome=reference,
    uniq=FALSE, extend=120, window_size=500,
    chr.select=c(paste0("chr", 1:22), "chrX", "chrY")
) )
}

#Differential Enrichment Analysis (takes about 30 minutes)
resNvPos=MEDIPS.meth(MSet1=MSet[["CCR4pos"]], MSet2=MSet[["Naive"]], diff.
method="edgeR", MeDIP=F, minRowSum=10)

#Selecting a set of highly significant DERs by applying a significance threshold to
the p-values adjusted for multiple testing:
sigNvPos=MEDIPS.selectSig(resNvPos, p.value=0.01, adj=T)

#Selecting a second set of less significant windows by applying a significant
threshold to the unadjusted p-values (for visualization purpose only, see the
MvA plot below):
trendNvPos= MEDIPS.selectSig(resNvPos, p.value=0.005, adj=F)
#Dividing the DERs into those that gain H3K4me2 in T_H2 cells and those that lose
H3K4me2 in T_H2 cells compared to naïve T-cells. Please note, log2 ratios are
reported as log2(MSet1/MSet2).
sigNvPosGain = sigNvPos[which(sigNvPos$edgeR.logFC>0),]
sigNvPosLoss = sigNvPos[which(sigNvPos$edgeR.logFC<0),]

#Merging neighboring DERs:
sigNvPosGainMerged = MEDIPS.mergeFrames(sigNvPosGain)
sigNvPosLossMerged = MEDIPS.mergeFrames(sigNvPosLoss)
```

```
#Annotating DERs by Ensembl transcript names, if the DERs are locate in a promoter
region (-1kb to +0.5kb around the transcription start sites):
tss_ens = MEDIPS.getAnnotation(host="www.biomart.org",
   dataset="hsapiens_gene_ensembl",
   annotation="TSS",tssSz=c(-1000,500))
sigNvPosGainMerged = MEDIPS.setAnnotation(sigNvPosGainMerged, tss_ens)
sigNvPosLossMerged = MEDIPS.setAnnotation(sigNvPosLossMerged, tss_ens)

#Create an MvA plot. This section creates the plot shown in Fig. 3, if the complete
data set has been processed (and not only the three example samples per
condition):
png("plots/MA_Comparison_Naive_vs_CCR4Pos.png",
   width=800, height=500)
smoothScatter(x=resNvPos$edgeR.logCPM,
   y=resNvPos$edgeR.logFC,
   xlim=c(-3.5, 4), ylim=c(-4.5, 4.5),
   pch=".", main="Genome wide 500bp windows",
   xlab="log2(average counts per million)",
   ylab="log2(CCR4+ / Naive)")
points(x=trendNvPos$edgeR.logCPM, y=trendNvPos$edgeR.logFC,
   pch=".", col="orange")
points(x=sigNvPos$edgeR.logCPM, y=sigNvPos$edgeR.logFC,
   pch=".", col="red")
abline(h=0, col="red")
dev.off()
```

Acknowledgements

This work is supported by the German Federal Ministry of Education and Research with the grant EPITREAT (No. 0316190A) and by the Max Planck Society with its International Research School program (IMPRS-CBSC) (to M.L.). L.C. is the recipient of a Feodor Lynen postdoctoral Research Fellowship granted by the Alexander von Humboldt Foundation.

References

1. Lara-Astiaso D, Weiner A, Lorenzo-Vivas E, Zaretsky I, Jaitin DA, David E et al (2014) Immunogenetics. Chromatin state dynamics during blood formation. Science 345(6199):943–949
2. Weber M, Davies JJ, Wittig D, Oakeley EJ, Haase M, Lam WL et al (2005) Chromosome-wide and promoter-specific analyses identify sites of differential DNA methylation in normal and transformed human cells. Nat Genet 37(8):853–862
3. Serre D, Lee BH, Ting AH (2010) MBD-isolated Genome Sequencing provides a high-throughput and comprehensive survey of DNA methylation in the human genome. Nucleic Acids Res 38(2):391–399
4. Pastor WA, Pape UJ, Huang Y, Henderson HR, Lister R, Ko M et al (2011) Genome-wide mapping of 5-hydroxymethylcytosine in embryonic stem cells. Nature 473 (7347):394–397

5. Zhang Y, Liu T, Meyer CA, Eeckhoute J, Johnson DS, Bernstein BE et al (2008) Model-based analysis of ChIP-Seq (MACS). Genome Biol 9(9):R137
6. Zang C, Schones DE, Zeng C, Cui K, Zhao K, Peng W (2009) A clustering approach for identification of enriched domains from histone modification ChIP-Seq data. Bioinformatics 25(15):1952–1958
7. Lienhard M, Grimm C, Morkel M, Herwig R, Chavez L (2014) MEDIPS: genome-wide differential coverage analysis of sequencing data derived from DNA enrichment experiments. Bioinformatics 30(2):284–286
8. Robinson MD, McCarthy DJ, Smyth GK (2010) edgeR: a Bioconductor package for differential expression analysis of digital gene expression data. Bioinformatics 26 (1):139–140
9. Seumois G, Chavez L, Gerasimova A, Lienhard M, Omran N, Kalinke L et al (2014) Epigenomic analysis of primary human T cells reveals enhancers associated with TH2 memory cell differentiation and asthma susceptibility. Nat Immunol 15(8):777–788
10. Langmead B, Trapnell C, Pop M, Salzberg SL (2009) Ultrafast and memory-efficient alignment of short DNA sequences to the human genome. Genome Biol 10(3):R25
11. Hnisz D, Abraham BJ, Lee TI, Lau A, Saint-Andre V, Sigova AA et al (2013) Super-enhancers in the control of cell identity and disease. Cell 155(4):934–947
12. Huang Y, Chavez L, Chang X, Wang X, Pastor WA, Kang J et al (2014) Distinct roles of the methylcytosine oxidases Tet1 and Tet2 in mouse embryonic stem cells. Proc Natl Acad Sci U S A 111(4):1361–1366