Analysis of ChIP-seq Data with '`mosaics`' Package: MOSAiCS and MOSAiCS-HMM

Dongjun Chung[1], Pei Fen Kuan[2], Rene Welch[3], and Sündüz Keleş[3,4]
[1]Department of Public Health Sciences, Medical University of South Carolina
Charleston, SC 29425.
[2]Department of Applied Mathematics and Statistics, Stony Brook University
Stony Brook, NY 11794.
[3]Department of Statistics, University of Wisconsin
Madison, WI 53706.
[4]Department of Biostatistics and Medical Informatics, University of Wisconsin
Madison, WI 53706.

May 9, 2023

Contents

1 Overview

This vignette provides an introduction to the analysis of ChIP-seq data with the '`mosaics`' package. R package `mosaics` implements MOSAiCS, a statistical framework for the analysis of ChIP-seq data, proposed in [1]. MOSAiCS stands for "**MO**del-based one and two **S**ample **A**nalysis and **I**nference for **C**hIP-**S**eq Data". It implements a flexible parametric mixture modeling approach for detecting peaks, i.e., enriched regions, in one-sample (ChIP sample) or two-sample (ChIP and matched control samples) ChIP-seq data. It can account for mappability and GC content biases that arise in ChIP-seq data.

Recently, we extended the MOSAiCS framework with a hidden Markov Model (HMM) architecture, named MOSAiCS-HMM [2], to identify broad peaks. By considering the HMM architecture, MOSAiCS-HMM allows modeling of read counts in histone modifiation ChIP-seq experiments and accounts for the spatial dependence in their ChIP-seq profiles.

The package can be loaded with the command:

```
R> library("mosaics")
```

'`mosaicsExample`' package is a companion package for the '`mosaics`' package and it provides various example ChIP-seq datasets for MOSAiCS and MOAiCS-HMM, as illustrated in this vignette.

```
R> library("mosaicsExample")
```

In this vignette, we focus on chromosome 22 data from ChIP-seq experiments of STAT1 binding and H3K4me3 modification in GM12878 cell line, downloaded from the ENCODE database (`http://genome.ucsc.edu/ENCODE/`). In Section 7, we consider chromosome 21 data from a ChIP-seq experiment of STAT1 binding in interferon-γ-stimulated HeLa S3 cells [3].

2 Getting started

'mosaics' package provides flexible framework for the ChIP-seq analysis. If you have the data for matched control sample, two-sample analysis is recommended. If the ChIP-seq data is deeply sequenced, the two-sample analysis without mappability and GC content (Section 3) is usually appropriate. For the ChIP-seq data with low sequencing depth, the two-sample analysis with mappability and GC content (Section 7) can be useful. When control sample is not available, 'mosaics' package accommodates one-sample analysis of ChIP-seq data. In this case, you should have files for mappability and GC content, in addition to the files for ChIP and matched control samples.

Section 3 discusses each step of the two-sample workflow and provides command lines for each step. Section 4 discusses steps of analyzing histone modification ChIP-seq experiments using MOSAiCS-HMM. Section 5 discusses steps of post-processing peak call results, including identifying peak summits, adjusting peak boundaries, and filtering potentionally false positive peaks. Section 6 discusses the most convenient way to do the two-sample analysis (without using mappability and GC content) by running a single function. Sections 7 and 8 briefly explain the workflow and command lines for the two-sample analysis and the one-sample analysis with mappability and GC content, respectively.

We encourage questions or requests regarding 'mosaics' package to be posted on our Google group http://groups.google.com/group/mosaics_user_group.

3 Identification of Punctuated Peaks using MOSAiCS

3.1 Constructing Bin-Level Files from the Aligned Read File

R package 'mosaics' analyzes the data after converting aligned read files into bin-level files for modeling and visualization purposes. Bin-level data for ChIP and control samples can easily be generated from the aligned read files with the command:

```
R> constructBins( infile=system.file( file.path("extdata","wgEncodeSydhTfbsGm12878Stat1StdAlnRep
+     fileFormat="bam", outfileLoc="./",
+     byChr=FALSE, useChrfile=FALSE, chrfile=NULL, excludeChr=NULL,
+     PET=FALSE, fragLen=200, binSize=200, capping=0 )

------------------------------------------------------------
Info: setting summary
------------------------------------------------------------
Name of aligned read file: /Library/Frameworks/R.framework/Versions/4.3-arm64/Resources/library/
Aligned read file format: BAM
Directory of processed bin-level files: ./
Construct bin-level files by chromosome? N
Is file for chromosome info provided? N
Data type: Single-end tag (SET)
Average fragment length: 200
Bin size: 200
------------------------------------------------------------
------------------------------------------------------------
Info: processing summary
------------------------------------------------------------
```

```
Directory of processed bin-level files: ./
Processed bin-level file: wgEncodeSydhTfbsGm12878Stat1StdAlnRep1_chr22_sorted.bam_fragL200_bin20
Sequencing depth: 231822
------------------------------------------------------------

R> constructBins( infile=system.file( file.path("extdata","wgEncodeSydhTfbsGm12878InputStdAlnRep
+     fileFormat="bam", outfileLoc="./",
+     byChr=FALSE, useChrfile=FALSE, chrfile=NULL, excludeChr=NULL,
+     PET=FALSE, fragLen=200, binSize=200, capping=0 )

------------------------------------------------------------
Info: setting summary
------------------------------------------------------------
Name of aligned read file: /Library/Frameworks/R.framework/Versions/4.3-arm64/Resources/library/
Aligned read file format: BAM
Directory of processed bin-level files: ./
Construct bin-level files by chromosome? N
Is file for chromosome info provided? N
Data type: Single-end tag (SET)
Average fragment length: 200
Bin size: 200
------------------------------------------------------------
------------------------------------------------------------
Info: processing summary
------------------------------------------------------------
Directory of processed bin-level files: ./
Processed bin-level file: wgEncodeSydhTfbsGm12878InputStdAlnRep1_chr22_sorted.bam_fragL200_bin20
Sequencing depth: 182605
------------------------------------------------------------
```

You can specify the name and file format of the aligned read file in '`infile`' and '`fileFormat`' arguments, respectively. The '`PET`' argument indicates whether the file is paired-end tag ('`PET=TRUE`') or single-end tag ('`PET=FALSE`') ChIP-Seq data. Allowed aligned read file formats depend on the '`PET`' argument. '`constructBins`' method currently allows the following aligned read file formats for SET ChIP-Seq data: BAM ('`"bam"`'), SAM ('`"sam"`'), BED ('`"bed"`'), Eland result ('`"eland_result"`'), Eland extended ('`"eland_extended"`'), Eland export ('`"eland_export"`'), default Bowtie ('`"bowtie"`'), and CSEM BED ('`"csem"`'). For PET ChIP-Seq data, '`constructBins`' method currently accepts the BAM ('`"bam"`'), SAM ('`"sam"`'), and Eland result ('`"eland_result"`') file formats. If input file format is neither BED nor CSEM BED, it retains only reads mapping uniquely to the reference genome (uni-reads). See Appendices A and B for example lines of each aligned read file format.

Even though '`constructBins`' retains only uni-reads for most aligned read file formats, reads mapping to multiple locations on the reference genome (multi-reads) can be easily incorporated into bin-level files by utilizing our multi-read allocator, `CSEM` (ChIP-Seq multi-read allocator using Expectation-Maximization algorithm). Galaxy tool for CSEM is also available in Galaxy Tool Shed (`http://toolshed.g2.bx.psu.edu/`; "csem" under "Next Gen Mappers") and standalone version of CSEM is also available at `http://deweylab.biostat.wisc.edu/csem/`. CSEM exports uni-reads and allocated multi-reads into standard BED file and the corresponding bin-level

files can be constructed by applying 'constructBins' method to this BED file with the argument 'fileFormat="csem"'.

'constructBins' can generate a single bin-level file containing all chromosomes (for a genome-wide analysis) or multiple bin-level files for each chromosome (for a chromosome-wise analysis). If 'byChr=FALSE', bin-level data for all chromosomes are exported to one file named as '[infileName]_fragL[fragLen (for SET data) or '[infileName]_bin[binSize].txt' (for PET data), where [infileName], [fragLen], and [binSize] are name of aligned read file, average fragment length, and bin size, respectively. If 'byChr=TRUE', bin-level data for each chromosome is exported to a separate file named as '[infileName]_fragL[fra (for SET data) or '[infileName]_bin[binSize]_[chrID].txt' (for PET data), where [chrID] is chromosome ID that reads align to. These chromosome IDs ([chrID]) are extracted from the aligned read file. The constructed bin-level files are exported to the directory specified in 'outfileLoc' argument.

The 'useChrfile' argument indicates whether to use the file containing chromosome ID and chromosome size, which is specified in the 'chrfile' argument. See Appendix C for the example lines of this chromosome information file. If you want to exclude some chromosomes in the processed bin-level files, you can specify these chromosomes in 'excludeChr' argument. The 'excludeChr' argument will be ignored if 'useChrfile=TRUE'.

You can specify average fragment length and bin size in 'fragLen' and 'binSize' arguments, respectively, and these arguments control the resolution of bin-level ChIP-seq data. By default, average fragment length is set to 200 bp, which is the common fragment length for Illumina sequences, and bin size equals to average fragment length. The 'fragLen' argument is ignored for PET ChIP-seq data ('PET = TRUE'). 'capping' argument indicates maximum number of reads allowed to start at each nucleotide position. Using some small value for capping (e.g., 'capping=3') will exclude extremely large read counts that might correspond to PCR amplification artifacts, which is especially useful for the ChIP-seq data with low sequencing depth. Capping is not applied (default) if 'capping' is set to some non-positive value, e.g., 'capping=0'.

3.2 Reading Bin-Level Data into the R Environment

You now have bin-level ChIP data and matched control sample data from 'constructBins'. Bin-level data can be imported to the R environment with the command:

```
R> fileName <- c(
+     "wgEncodeSydhTfbsGm12878Stat1StdAlnRep1_chr22_sorted.bam_fragL200_bin200.txt",
+     "wgEncodeSydhTfbsGm12878InputStdAlnRep1_chr22_sorted.bam_fragL200_bin200.txt")
R> binTFBS <- readBins( type=c("chip","input"), fileName=fileName )
```

For the 'type' argument, "chip" and "input" indicate bin-level ChIP data control sample data, respectively. You need to specify the corresponding file names in 'fileName'. 'mosaics' package assumes that each file name in 'fileName' is provided in the same order as in 'type'.

In mosaics package, you can do either genome-wide analysis or chromosome-wise analysis and this analysis type will be determined automatically based on the contents of bin-level files imported using 'readBins'. If the bin-level files contain more than one chromosome (i.e., bin-level files are obtained using 'byChr=FALSE' in 'constructBins'), 'mosaicsFit' will analyze all the chromosomes simultaneously (genome-wide analysis). Note that if these bin-level files contain different sets of chromosomes, then 'readBins' method will utilize only the intersection of them. If bin-level files are obtained using 'byChr=FALSE' in 'constructBins', each bin-level file contains data for only one chromosome and each of these bin-level files need to be analyzed separately (chromosome-

wise analysis). The genome-wide analysis usually provide more stable model fitting and peak identification results so it is recommended for most cases.

R package `mosaics` provides functions for generating simple summaries of the data. The following command prints out basic information about the bin-level data, such as number of bins and total "effective tag counts". "Total effective tag counts" is defined as the sum of the ChIP tag counts of all bins. This value is usually larger than the sequencing depth since tags are counted after extension to average fragment length and an extended fragment can contribute to multiple bins.

```
R> binTFBS

Summary: bin-level data (class: BinData)
----------------------------------------
- # of chromosomes in the data: 1
- total effective tag counts: 462479
  (sum of ChIP tag counts of all bins)
- control sample is incorporated
- mappability score is NOT incorporated
- GC content score is NOT incorporated
- uni-reads are assumed
----------------------------------------
```

'`print`' method returns the bin-level data in data frame format.

```
R> print(binTFBS)[ 90600:90614, ]

      chrID    coord tagCount input
90600 chr22 18119800        0     2
90601 chr22 18120000        0     2
90602 chr22 18120200        2     1
90603 chr22 18120400        3     0
90604 chr22 18120600        2     2
90605 chr22 18120800        5     6
90606 chr22 18121000        4     7
90607 chr22 18121200       14    10
90608 chr22 18121400       27    13
90609 chr22 18121600       16    10
90610 chr22 18121800        4     6
90611 chr22 18122000        7     9
90612 chr22 18122200        6    16
90613 chr22 18122400        3    12
90614 chr22 18122600        2     3
```

'`plot`' method provides exploratory plots for the ChIP data. Different type of plots can be obtained by varying the '`plotType`' argument. '`plotType="input"`' generates a plot of mean ChIP tag counts versus control tag counts. If '`plotType`' is not specified, this method plots the histogram of ChIP tag counts.

```
R> plot( binTFBS )
R> plot( binTFBS, plotType="input" )
```

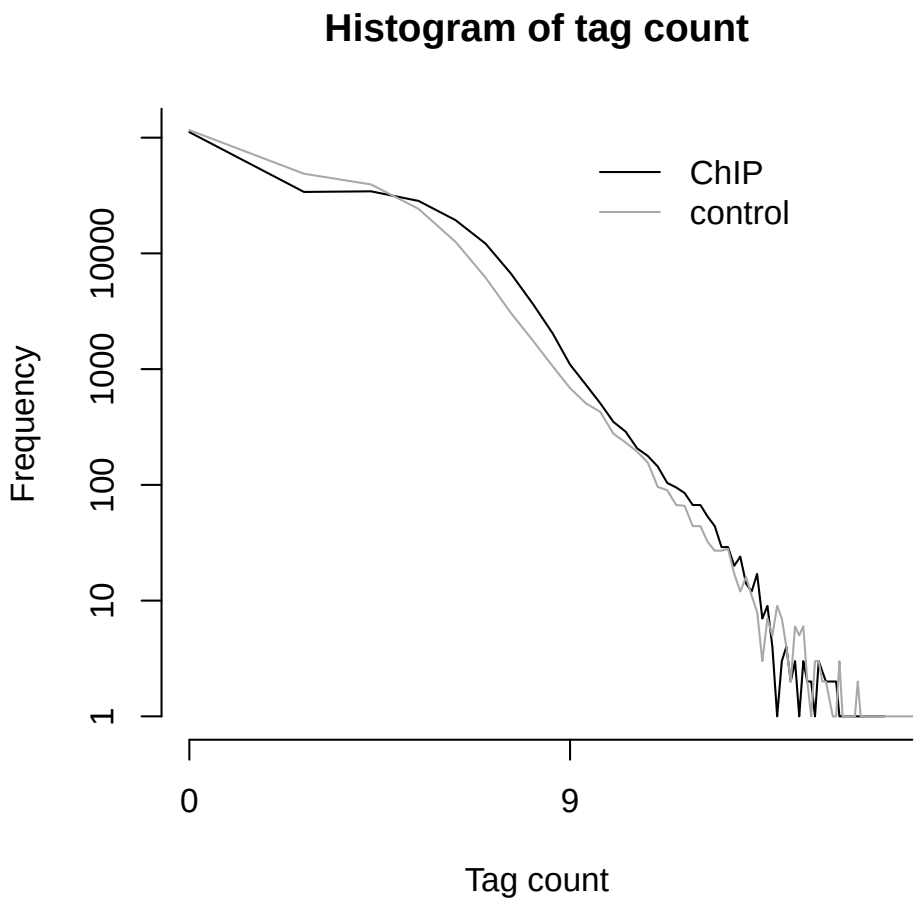


Figure 1: Histograms of the count data from ChIP and control samples.

Figures 1 and 2 display examples of different types of plots. The relationship between mean ChIP tag counts and control tag counts seems to be linear, especially for small control tag counts (Figure 2).

3.3 Fitting the MOSAiCS Model

We are now ready to fit a MOSAiCS model using the bin-level data above (`binTFBS`) with the command:

```
R> fitTFBS <- mosaicsFit( binTFBS, analysisType="IO", bgEst="rMOM" )
```

'`analysisType="IO"`' indicates implementation of the two-sample analysis. '`bgEst`' argument determines background estimation approach. '`bgEst="matchLow"`' estimates background distribution using only bins with low tag counts and it is appropriate for the data with relatively low sequencing depth. '`bgEst="rMOM"`' estimates background distribution using robust method of moment (MOM) (default) and it is appropriate for the data with relatively high sequencing depth. If '`bgEst="automatic"`', '`mosaicsFit`' tries its best guess for the background estimation approach,

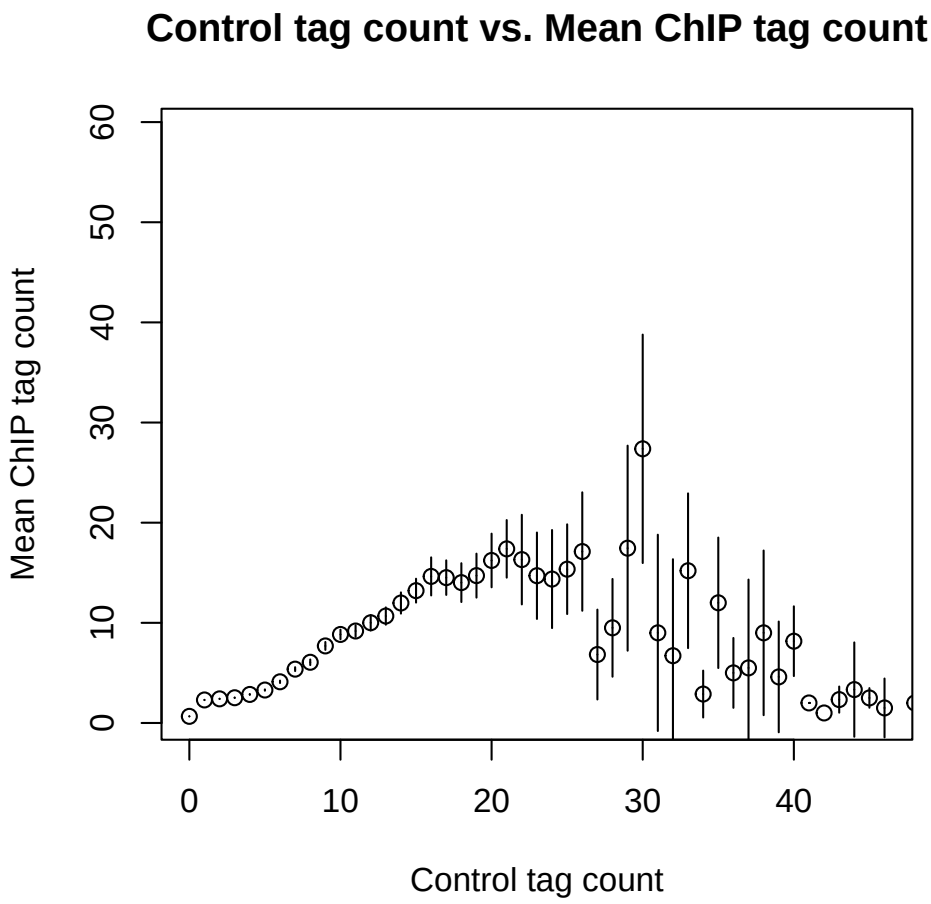


Figure 2: Mean ChIP tag count versus Control tag count.

based on the data provided. If the goodness of fit obtained using '`bgEst="automatic"`' is not satisfactory, we recommend users to try '`bgEst="matchLow"`' and '`bgEst="rMOM"`' and it might improve the model fit.

'`mosaicsFit`' fits both one-signal-component and two-signal-component models. When identifying peaks, you can choose the number of signal components to be used for the final model. The optimal choice of the number of signal components depends on the characteristics of data. In order to support users in the choice of optimal signal model, `mosaics` package provides Bayesian Information Criterion (BIC) values and Goodness of Fit (GOF) plots of these signal models.

The following command prints out BIC values of one-signal-component and two-signal-component models, with additional information about the parameters used in fitting the background (non-enriched) distribution. A lower BIC value indicates a better model fit. In this dataset, BIC values are quite comparable between one- and two-signal-component models although BIC value is slightly smaller for one-signal-component model.

```
R> fitTFBS

Summary: MOSAiCS model fitting (class: MosaicsFit)
```

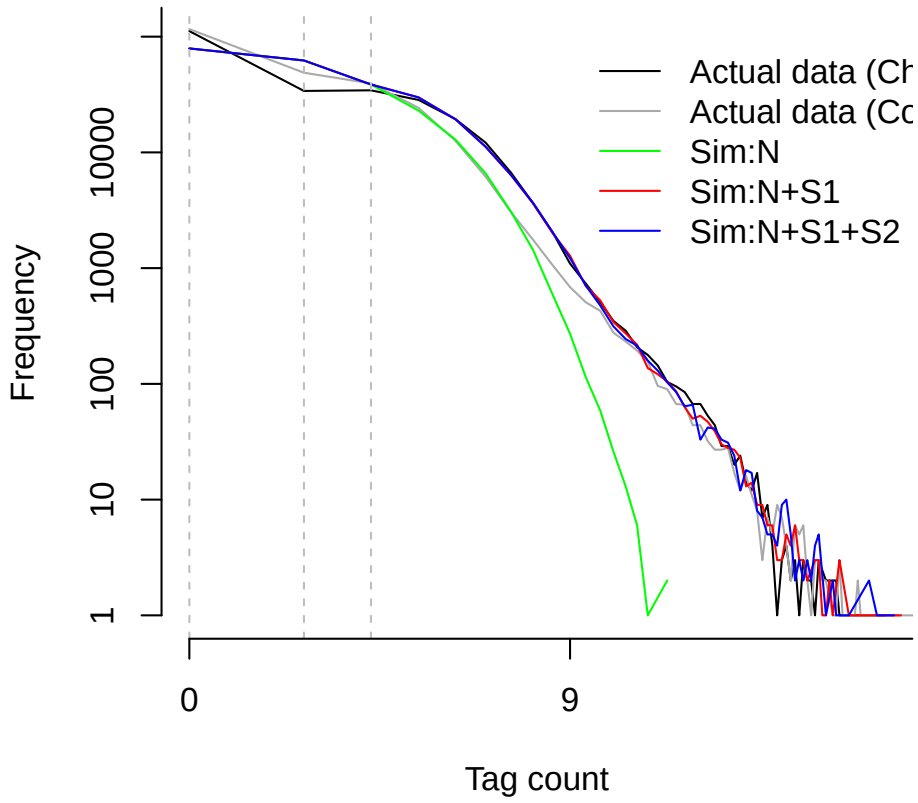


Figure 3: Goodness of Fit (GOF) plot. Depicted are actual data for ChIP and control samples with simulated data from the following fitted models: (Sim:N): Background model; (Sim:N+S1): one-signal-component model; (Sim:N+S1+S2): two-signal-component model.

```
-------------------------------------------------
analysis type: two-sample analysis (Input only)
parameters used: k = 3, d = 0.25
BIC of one-signal-component model = 861725.3
BIC of two-signal-component model = 861764.2
-------------------------------------------------
```

'`plot`' method provides the GOF plot. This plots allows visual comparisons of the fits of the background, one-signal-component, and two-signal-component models with the actual data. Figure 3 displays the GOF plot for our dataset and we conclude that the one- and two-signal-component models provide comparable model fit as is also suggested by BIC values.

```
R> plot(fitTFBS)
```

3.4 Identifying Peaks Based on the Fitted Model

We can now identify peaks with the two-signal-component model at a false discovery rate (FDR) of 0.05 with the command:

```
R> peakTFBS <- mosaicsPeak( fitTFBS, signalModel="2S", FDR=0.05,
+ maxgap=200, minsize=50, thres=10 )
```

'`signalModel="2S"`' indicates two-signal-component model. Similarly, one-signal-component model can be specified by '`signalModel="1S"`'. FDR can be controlled at the desired level by specifying '`FDR`' argument. In addition to these two essential parameters, you can also control three more parameters, '`maxgap`', '`minsize`', and '`thres`'. These parameters are for refining initial peaks called using specified signal model and FDR. Initial nearby peaks are merged if the distance (in bp) between them is less than '`maxgap`'. Some initial peaks are removed if their lengths are shorter than '`minsize`' or their ChIP tag counts are less than '`thres`'.

If you use a bin size shorter than the average fragment length in the experiment, we recommend to set '`maxgap`' to the average fragment length and '`minsize`' to the bin size. This setting removes peaks that are too narrow (e.g., singletons). If you set the bin size to the average fragment length (or maybe bin size is larger than the average fragment length), we recommend setting '`minsize`' to a value smaller than the average fragment length while leaving '`maxgap`' the same as the average fragment length. This is to prevent filtering using '`minsize`' because initial peaks would already be at a reasonable width. '`thres`' is employed to filter out initial peaks with very small ChIP tag counts because such peaks might be false discoveries. Optimal choice of '`thres`' depends on the sequencing depth of the ChIP-seq data to be analyzed. If you don't wish to filter out initial peaks using ChIP tag counts, you can set '`thres`' to an arbitrary negative value.

The following command prints out a summary of identified peaks including the number of peaks identified, median peak width, and the empirical false discovery rate (FDR).

```
R> peakTFBS
```

```
Summary: MOSAiCS peak calling (class: MosaicsPeak)
-------------------------------------------------
final model: two-sample analysis (input only) with two signal components
setting: FDR = 0.05, maxgap = 200, minsize = 50, thres = 10
# of peaks = 774
median peak width = 400
empirical FDR = 0.0517
read-level data is currently not loaded.
[Note] Read-level data is needed to run findSummit(), adjustBoundary(), and filterPeak().
[Note] Run extractReads() to load read-level data.
-------------------------------------------------
```

'`print`' method returns the peak calling results in data frame format. This data frame can be used as an input for downstream analysis such as motif finding. This output might have different number of columns, depending on '`analysisType`' of '`mosaicsFit`'. For example, in the case of two-sample analysis ('`analysisType="IO"`'), columns are chromosome ID, peak start position, peak end position, peak width, averaged posterior probability, minimum posterior probability, averaged ChIP tag count, maximum ChIP tag count, averaged control tag count, averaged control tag count scaled by sequencing depth, and averaged log base 2 ratio of ChIP over input tag counts for each peak.

Here, the posterior probability of a bin refers to the probability that the bin belongs to background conditional on data. Hence, smaller posterior probabilities provide more evidence that the bin is actually a peak.

```
R> head(print(peakTFBS))
```

```
  chrID peakStart peakStop peakSize  logAveP   logMinP  aveLogP aveChipCount
1 chr22  16848400 16848999      600 2.078949  2.858803 2.339062     25.00000
2 chr22  16861600 16862199      600 4.269255 13.851900 7.710193     42.66667
3 chr22  17255600 17255799      200 3.075094  3.075094 3.075094     12.00000
4 chr22  17306200 17306599      400 2.285716  2.313910 2.286576     13.00000
5 chr22  17539000 17539399      400 1.911207  2.995591 2.312014     17.50000
6 chr22  17568800 17568999      200 1.423776  1.423776 1.423776     13.00000
  maxChipCount aveInputCount aveInputCountScaled aveLog2Ratio
1           29      20.00000           25.390542 -0.008591853
2           59      72.66667           92.252304 -1.129960279
3           12       1.00000            1.269527  2.518047989
4           14       2.50000            3.173818  1.759177555
5           20       8.00000           10.156217  0.725784753
6           13       5.00000            6.347636  0.930074842
```

You can export peak calling results to text files in diverse file formats. Currently, '`mosaics`' package supports TXT, BED, GFF, narrowPeak, and broadPeak file formats. In the case of TXT file format ('`type="txt"`'), all the columns provided by '`print`' method are exported. '`type="bed"`' and '`type="gff"`' export peak calling results in standard BED and GFF file formats, respectively, where score is the averaged ChIP tag counts in each peak. narrowPeak and broadPeak are modified BED file formats and these file formats are used by the ENCODE Consortium and more information about these file formats can be found in `http://genome.ucsc.edu/FAQ/FAQformat.html`. In narrowPeak and broadPeak file formats, the 5th, 7th, and 8th columns are averaged log base 2 ratio of ChIP over input tag counts (or averaged ChIP tag counts, if the matched control sample is not provided), ChIP signal at peak summit, and -log10 transformed minimum posterior probability. Note that narrowPeak and broadPeak file formats are supported only after '`extractReads`' and '`findSummit`' methods are applied to '`MosaicsPeak`' object because it requires peak summit information. See Section 5.1 for more information.

Peak calling results can be exported in TXT, BED, GFF, narrowPeak, and broadPeak file formats, respectively, by the following commands, were '`fileLoc`' and '`fileName`' indicate the directory and the name of the exported file:

```
R> export( peakTFBS, type="txt", filename="peakTFBS.txt" )
R> export( peakTFBS, type="bed", filename="peakTFBS.bed" )
R> export( peakTFBS, type="gff", filename="peakTFBS.gff" )
R> export( peakTFBS, type="narrowPeak", filename="peakTFBS.narrowPeak" )
R> export( peakTFBS, type="broadPeak", filename="peakTFBS.broadPeak" )
```

3.5 Generating Wiggle Files to View on Genome Browsers

R package '`mosaics`' can generate wiggle files (`http://genome.ucsc.edu/goldenPath/help/wiggle.html`) for visualization purposes. These wiggle files can be used as input for many genome browsers such as the UCSC genome browser (`http://genome.ucsc.edu/`) and IGV (`http://www.broadinstitute.org/igv/`). These wiggle files can easily be generated from the aligned read files with the command:

```
R> generateWig( infile=system.file( file.path("extdata","wgEncodeSydhTfbsGm12878Stat1StdAlnRep1_
+     fileFormat="bam", outfileLoc="./",
+     byChr=FALSE, useChrfile=FALSE, chrfile=NULL, excludeChr=NULL,
+     PET=FALSE, fragLen=200, span=200, capping=0, normConst=1 )

------------------------------------------------------------
Info: setting summary
------------------------------------------------------------
Name of aligned read file: /Library/Frameworks/R.framework/Versions/4.3-arm64/Resources/library/
Aligned read file format: BAM
Directory of processed wig files: ./
span of the wig files: 200
Normalizing constant: 1
Construct wig files by chromosome? N
Is file for chromosome info provided? N
Data type: Single-end tag (SET)
Average fragment length: 200
------------------------------------------------------------
------------------------------------------------------------
Info: processing summary
------------------------------------------------------------
Directory of processed wig files: ./
Processed wig file: wgEncodeSydhTfbsGm12878Stat1StdAlnRep1_chr22_sorted.bam_fragL200_span200.wig
------------------------------------------------------------
```

The '`generateWig`' function has similar arguments as the '`constructBins`' function, except that it has '`span`' and '`normConst`' arguments instead of the '`binSize`' argument of the '`constructBins`' function. The '`generateWig`' function supports the same set of aligned read file formats as in the '`constructBins`' function. The '`span`' argument indicates span used in wiggle files. The '`normConst`' argument means the normalizing constant to scale the values in wiggle files and it is especially useful when wiggle files for multiple related samples are generated and compared. In the resulting wiggle files, values are scaled by the value specified in the '`normConst`' argument.

'`generateWig`' can generate a single wiggle file containing all chromosomes or multiple wiggle files for each chromosome. If '`byChr=FALSE`', all chromosomes are exported to one file named as '`[infileName]_fragL[fragLen]_span[span].wig`' (for SET data) or '`[infileName]_span[span].wig`' (for PET data), where `[infileName]`, `[fragLen]`, and `[span]` are name of aligned read file, average fragment length, and span, respectively. If '`byChr=TRUE`', each chromosome is exported to a separate file named as '`[infileName]_fragL[fragLen]_span[span]_[chrID].wig`' (for SET data) or '`[infileName]_span[span]_[chrID].wig`' (for PET data), where `[chrID]` is chromosome ID that reads align to. The constructed wiggle files are exported to the directory specified in '`outfileLoc`' argument.

4 Identification of Broad Peaks using MOSAiCS-HMM

In '`mosaics`' package, broad peaks can be identified with MOSAiCS-HMM model, using methods '`mosaicsFitHMM`' and '`mosaicsPeakHMM`'. In this section, we use ChIP-seq data of H3K4me3 for illustration. For computational efficiency, '`mosaicsFitHMM`' utilizes MOSAiCS model fit as estimates of emission distribution of MOSAiCS-HMM model. In addition, it also considers MOSAiCS peak

calling results at specified FDR level (‘`init.FDR`’) as initial values by default. MOSAiCS model fit for H3K4me3 data can be obtained with the commands:

```
R> constructBins( infile=system.file( file.path("extdata","wgEncodeBroadHistoneGm12878H3k4me3Std
+     fileFormat="bam", outfileLoc="./",
+     byChr=FALSE, useChrfile=FALSE, chrfile=NULL, excludeChr=NULL,
+     PET=FALSE, fragLen=200, binSize=200, capping=0 )

------------------------------------------------------------
Info: setting summary
------------------------------------------------------------
Name of aligned read file: /Library/Frameworks/R.framework/Versions/4.3-arm64/Resources/library/
Aligned read file format: BAM
Directory of processed bin-level files: ./
Construct bin-level files by chromosome? N
Is file for chromosome info provided? N
Data type: Single-end tag (SET)
Average fragment length: 200
Bin size: 200
------------------------------------------------------------
------------------------------------------------------------
Info: processing summary
------------------------------------------------------------
Directory of processed bin-level files: ./
Processed bin-level file: wgEncodeBroadHistoneGm12878H3k4me3StdAlnRep1_chr22_sorted.bam_fragL200
Sequencing depth: 357163
------------------------------------------------------------

R> constructBins( infile=system.file( file.path("extdata","wgEncodeBroadHistoneGm12878ControlStd
+     fileFormat="bam", outfileLoc="./",
+     byChr=FALSE, useChrfile=FALSE, chrfile=NULL, excludeChr=NULL,
+     PET=FALSE, fragLen=200, binSize=200, capping=0 )

------------------------------------------------------------
Info: setting summary
------------------------------------------------------------
Name of aligned read file: /Library/Frameworks/R.framework/Versions/4.3-arm64/Resources/library/
Aligned read file format: BAM
Directory of processed bin-level files: ./
Construct bin-level files by chromosome? N
Is file for chromosome info provided? N
Data type: Single-end tag (SET)
Average fragment length: 200
Bin size: 200
------------------------------------------------------------
------------------------------------------------------------
Info: processing summary
------------------------------------------------------------
Directory of processed bin-level files: ./
```

```
Processed bin-level file: wgEncodeBroadHistoneGm12878ControlStdAlnRep1_chr22_sorted.bam_fragL200
Sequencing depth: 85165
-----------------------------------------------------------
```

```
R> fileName <- c(
+     "wgEncodeBroadHistoneGm12878H3k4me3StdAlnRep1_chr22_sorted.bam_fragL200_bin200.txt",
+     "wgEncodeBroadHistoneGm12878ControlStdAlnRep1_chr22_sorted.bam_fragL200_bin200.txt")
R> binHM <- readBins( type=c("chip","input"), fileName=fileName)
R> fitHM <- mosaicsFit( binHM, analysisType="IO", bgEst="rMOM" )
```

By taking the MOSAiCS model fit from '`mosaicsFit`' as an input, users can fit MOSAiCS-HMM with the command:

```
R> hmmHM <- mosaicsFitHMM( fitHM, signalModel = "2S",
+         init="mosaics", init.FDR = 0.05, parallel=FALSE, nCore=8 )
```

The following command prints out BIC values of MOSAiCS and MOSAiCS-HMM models so that users can determine whether it is beneficial to consider broad peaks using the MOSAiCS-HMM model. As before, a lower BIC value indicates a better model fit and in this H3K4me3 example, it is beneficial to consider the MOSAiCS-HMM model.

```
R> hmmHM
```

```
Summary: MOSAiCS-HMM model fitting (class: MosaicsHMM)
-----------------------------------------------------
final model: two-sample analysis (input only) with two signal components
setting for initialization:
Initialized using MOSAiCS peak calling results.
FDR = 0.05, maxgap = 200, minsize = 50, thres = 10
BIC of MOSAiCS model = 988738
BIC of MOSAiCS-HMM model = 911887.3
-----------------------------------------------------
```

'`plot`' method also provides the GOF plot, which allows visual comparisons of the fits of the background, MOSAiCS, and MOSAiCS-HMM models with the actual data. Figure 4 displays the GOF plot for our dataset.

```
R> plot(hmmHM)
```

'`mosaicsPeakHMM`' calls peaks based on the MOSAiCS-HMM model fit. '`mosaicsPeakHMM`' allows two approaches, Viterbi algorithm ('`decoding="viterbi"`') and posterior decoding ('`decoding="posterior"`'), to call peaks. When posterior decoding is used (default), users can specify FDR level for the posterior decoding in the argument '`FDR`'. '`mosaicsPeakHMM`' provides its output as '`MosaicsPeak`' class object, which '`mosaicsPeak`' also generates. Hence, all the methods used for the output of '`mosaicsPeak`' can also be used for the output of '`mosaicsPeakHMM`', e.g., '`export`', '`print`', and '`show`'. Users can call peaks based on the MOSAiCS-HMM model fit with the command:

```
R> peakHM <- mosaicsPeakHMM( hmmHM, FDR = 0.05, decoding="posterior",
+   thres=10, parallel=FALSE, nCore=8 )
```

Note that '`mosaicsFitHMM`' and '`mosaicsPeakHMM`' fits MOSAiCS-HMM model and calls peaks for each chromosome separately. Hence, whenever multiple cores are available, it is strongly recommended to utilize parallel processing by setting '`parallel=TRUE`' and specifying the number of available cores in the argument '`nCore`'.

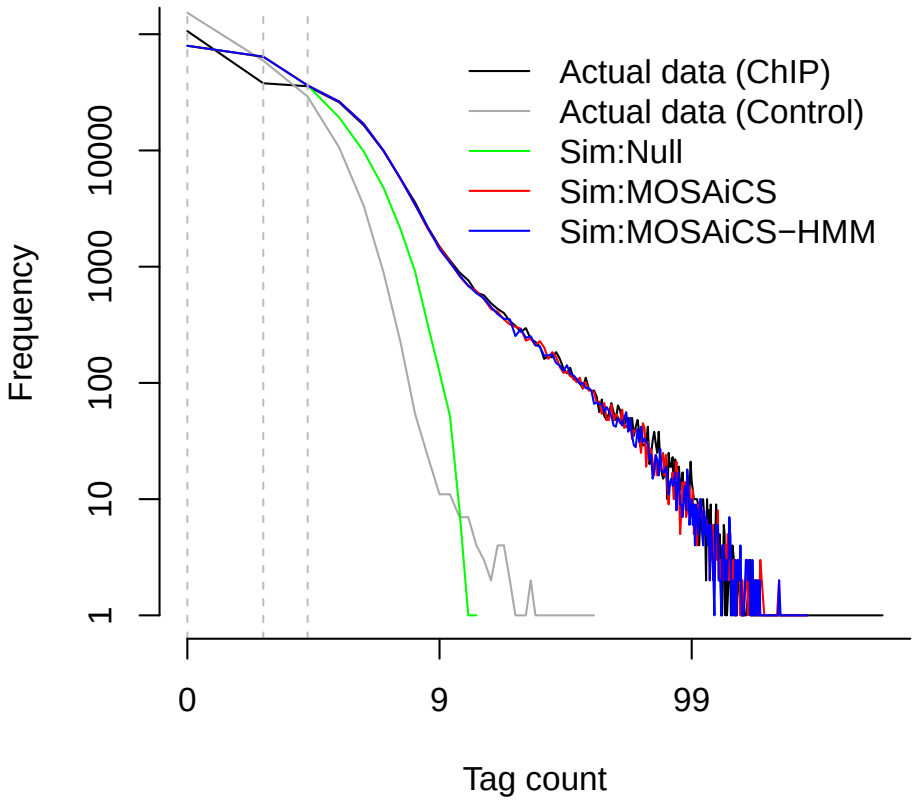


Figure 4: Goodness of Fit (GOF) plot. Depicted are actual data for ChIP and control samples with simulated data from the following fitted models: (Sim:Null): Background model; (Sim:MOSAiCS): MOSAiCS model; (Sim:MOSAiCS-HMM): MOSAiCS-HMM model.

5 Post-processing Steps after Peak Calling

5.1 Identification of Peak Summits

In the case of punctuated peaks, such as regions bound by transcription factors, a peak summit (i.e., genomic position with the highest ChIP profile) can potentially be considered as exact location of protein-DNA interaction. '`mosaics`' package provides functionality to identify a peak summit for each peak region identified using '`mosaicsPeak`' or '`mosaicsPeakHMM`' methods.

As '`mosaics`' package uses only information from bin-level data until this step, users first need to incorporate read-level data to identify peak summits. In this section, we consider H3K4me3 data as an example because this read-level data will also be needed in next sections discussing broad peaks. Read-level data can be loaded and incorporated into '`MosaicsPeak`' object, by applying '`extractReads`' method to '`MosaicsPeak`' object. The '`extractReads`' function supports the same set of aligned read file formats supported in the '`constructBins`' function.

```
R> peakHM <- extractReads( peakHM,
+   chipFile=system.file( file.path("extdata","wgEncodeBroadHistoneGm12878H3k4me3StdAlnRep1_chr2
+   chipFileFormat="bam", chipPET=FALSE, chipFragLen=200,
+   controlFile=system.file( file.path("extdata","wgEncodeBroadHistoneGm12878ControlStdAlnRep1_c
+   controlFileFormat="bam", controlPET=FALSE, controlFragLen=200, parallel=FALSE, nCore=8 )

------------------------------------------------------------
Info: Preprocessing summary
------------------------------------------------------------
Number of chromosomes: 1
Number of peaks: 2424
ChIP sample:
        Tag type: SET
        Sequencing depth: 357163
        Number of utilized reads: 161881
        Median number of reads in each peak: 17
Matched control sample:
        Tag type: SET
        Sequencing depth: 85165
        Number of utilized reads: 8230
        Median number of reads in each peak: 2
------------------------------------------------------------

R> peakHM

Summary: MOSAiCS peak calling (class: MosaicsPeak)
-------------------------------------------------
final model: two-sample analysis (input only) with two signal components
setting: FDR = 0.05, maxgap = 0, minsize = 0, thres = 10
# of peaks = 2424
median peak width = 400
empirical FDR = 0.05
read-level data is loaded.
ChIP sample:
        Sequencing depth: 357163
        Number of utilized reads: 161881
        Median number of reads in each peak: 17
Matched control sample:
        Sequencing depth: 85165
        Number of utilized reads: 8230
        Median number of reads in each peak: 2
-------------------------------------------------
```

Users can now identify peak summits by applying '`findSummit`' method to '`MosaicsPeak`' object:

```
R> peakHM <- findSummit( peakHM, parallel=FALSE, nCore=8 )
```

Locations of peak summits are now incorporated as the last column in the peak list:

```
R> head(print(peakHM))
```

```
  chrID peakStart peakStop peakSize   logAveP   logMinP   aveLogP aveChipCount
1 chr22  16096200 16096599      400 3.7903934 8.1942797 5.8418259     10.00000
2 chr22  16106800 16106999      200 4.0030168 4.0030168 4.0030168     10.00000
3 chr22  16114600 16114999      400 5.5319478 9.2650261 7.2479920     17.50000
4 chr22  16115800 16115999      200 0.8756994 0.8756994 0.8756994     10.00000
5 chr22  16143400 16143799      400 2.0867352 2.4430714 2.1683708     11.50000
6 chr22  16145400 16145999      600 3.3436370 4.4379163 3.6025819     11.66667
  maxChipCount aveInputCount aveInputCountScaled aveLog2Ratio summitSignal
1           10      0.500000            2.096888    2.2710396            9
2           10      1.000000            4.193777    1.0826476            5
3           19      1.500000            6.290665    1.4009288           18
4           10      3.000000           12.581330   -0.3041213            2
5           12      4.000000           16.775107   -0.3273922            7
6           13      2.666667           11.183405    0.2250343            9
    summit
1 16096475
2 16106921
3 16114752
4 16115893
5 16143537
6 16145506
```

5.2 Adjust Peak Boundaries and Filter Out Potentially False Positive Peaks

While '`mosaicsPeakHMM`' provides initial peak calling based on the MOSAiCS-HMM model, we found that these peak calling results can be improved further by post-processing (boundary adjustment and peak filtering). Note that we developed and tested these post-processing steps mainly for broad peaks of histone modifications. While these functions can also be applied to punctuated peaks of transcription factors without any restrictions, parameters for these functions might not be optimal for punctuated peaks.

'`adjustBoundary`' method checks the ChIP read counts and the ratio of ChIP signal over matched control signal at boundaries and trims/extends peak boundaries to obtain more accurate boundaries:

```
R> peakHM <- adjustBoundary( peakHM, parallel=FALSE, nCore=8 )

------------------------------------------------------------
Info: peak boundary adjustment summary
------------------------------------------------------------
# all peaks:  2424
# peaks with trimmed boundaries:  842
# peaks with extended boundaries:  0
# peaks of which summits changed:  3
normC:  4.193777
------------------------------------------------------------

R> peakHM

Summary: MOSAiCS peak calling (class: MosaicsPeak)
------------------------------------------------
```

```
final model: two-sample analysis (input only) with two signal components
setting: FDR = 0.05, maxgap = 0, minsize = 0, thres = 10
# of peaks = 2424
median peak width = 400
empirical FDR = 0.05
read-level data is loaded.
ChIP sample:
        Sequencing depth: 357163
        Number of utilized reads: 161881
        Median number of reads in each peak: 17
Matched control sample:
        Sequencing depth: 85165
        Number of utilized reads: 8230
        Median number of reads in each peak: 2
-------------------------------------------------
```

R> head(print(peakHM))

```
  chrID peakStart peakStop peakSize   logAveP   logMinP   aveLogP aveChipCount
1 chr22  16096200 16096599      400 3.7903934 8.1942797 5.8418259     10.00000
2 chr22  16106800 16106999      200 4.0030168 4.0030168 4.0030168     10.00000
3 chr22  16114740 16114898      159 5.5319478 9.2650261 7.2479920     17.50000
4 chr22  16115800 16115999      200 0.8756994 0.8756994 0.8756994     10.00000
5 chr22  16143400 16143799      400 2.0867352 2.4430714 2.1683708     11.50000
6 chr22  16145400 16145999      600 3.3436370 4.4379163 3.6025819     11.66667
  maxChipCount aveInputCount aveInputCountScaled aveLog2Ratio summitSignal
1           10      0.500000            2.096888    2.2710396            9
2           10      1.000000            4.193777    1.0826476            5
3           19      1.500000            6.290665    1.4009288           18
4           10      3.000000           12.581330   -0.3041213            2
5           12      4.000000           16.775107   -0.3273922            7
6           13      2.666667           11.183405    0.2250343            9
    summit
1 16096475
2 16106921
3 16114752
4 16115893
5 16143537
6 16145506
```

Its summary output indicates that boundaries of 842 peaks are trimmed among 2,424 peaks called using '`mosaicsPeakHMM`' method. As such peak boundary adjustment can also affect peak summit identification (note that a peak summit should be located within its peak boundaries), peak summits are re-calculated after peak boundaries are adjusted. In this example, locations of 155 peak summits are changed by the peak boundary adjustment procedure.

'`filterPeak`' further checks the ChIP signal and the ratio of ChIP signal over matched control signal at the peak summit, in addition to the peak length, and filters out potentially false positive peaks:

```
R> peakHM <- filterPeak( peakHM, parallel=FALSE, nCore=8 )

------------------------------------------------------------
Info: peak filtering summary
------------------------------------------------------------
# peaks before filtering:  2424
# peaks after filtering #1:  1359
# peaks after filtering #2:  487
summitCut:  10
------------------------------------------------------------

R> peakHM

Summary: MOSAiCS peak calling (class: MosaicsPeak)
------------------------------------------------
final model: two-sample analysis (input only) with two signal components
setting: FDR = 0.05, maxgap = 0, minsize = 0, thres = 10
# of peaks = 487
median peak width = 1470
empirical FDR = 0.05
read-level data is loaded.
ChIP sample:
        Sequencing depth: 357163
        Number of utilized reads: 161881
        Median number of reads in each peak: 17
Matched control sample:
        Sequencing depth: 85165
        Number of utilized reads: 8230
        Median number of reads in each peak: 2
------------------------------------------------

R> head(print(peakHM))

   chrID peakStart peakStop peakSize   logAveP  logMinP  aveLogP aveChipCount
48 chr22  17084009 17085421     1413  4.205804 33.41119 14.46821     20.37500
67 chr22  17518273 17520583     2311  5.712321 26.83907 16.14161     26.33333
71 chr22  17565452 17569091     3640 11.088797 47.34735 21.64692     28.36842
77 chr22  17638793 17641093     2301  6.661809 51.88696 28.92271     41.69231
78 chr22  17652220 17652805      586 13.546699 18.53420 16.05901     29.75000
79 chr22  17698058 17700929     2872 10.237500 59.04056 28.64970     39.00000
   maxChipCount aveInputCount aveInputCountScaled aveLog2Ratio summitSignal
48           32      1.000000            4.193777     2.577915           22
67           41      1.500000            6.290665     2.181388           28
71           43      1.263158            5.297402     3.085905           32
77           70      1.153846            4.838973     2.887382           49
78           34      1.750000            7.339109     1.916722           25
79           54      1.333333            5.591702     3.040588           42
     summit
48 17084176
```

```
67 17519286
71 17568952
77 17640778
78 17652355
79 17700550
```

'filterPeak' employs two-step approach for peak filtering and its summary shows the peak filtering results after each step. The summary indicates that about half of the initial peaks (i.e., called using 'mosaicsPeakHMM') are filtered out after the first filtering step. Among the remaining peaks, about half of the peaks are further filtered out in the second step. After the peak filtering step using the 'filterPeak' method, we have 434 peaks for H3K4me3.

5.3 Visualization of ChIP Profile of Peak Regions

It is important to check ChIP profile to confirm that we have reasonable peak calling results. While mosaics package provides functionality to generate wiggle files for the use with genome browsers (See Section 3.5 for more details), users can also generate ChIP profile plots of peak regions directly from mosaics package for quick inspection of peak calling results.

In order to generate ChIP profile plots of peak regions, users first need to load read-level data using 'extractReads' method, as illustrated in Section 5.1. If 'extractReads' method is already applied to 'MosaicsPeak' object, users can generate ChIP profile plots of all the peak regions and save them as a single PDF file named 'peakplot.pdf' by the command:

```
R> plot( peakHM, filename="./peakplot.pdf" )
```

If users are interested only in some specific peak regions, users can specify peak regions to plot their ChIP profiles with the argument 'peakNum'. The input to argument 'peakNum' can be a vector, where elements indicate the row indexes in the peak list. For example, the 8-th peak of H3K4me3 is located at chr22:18120496-18122661 and 18-th peak of STAT1 factor overlaps this H3K4me3 peak:

```
R> print(peakHM)[8,]

   chrID peakStart peakStop peakSize  logAveP  logMinP  aveLogP aveChipCount
83 chr22  17738847 17739929     1083 12.69655 37.16249 21.74119         23.5
   maxChipCount aveInputCount aveInputCountScaled aveLog2Ratio summitSignal
83           35     0.1666667           0.6989628     4.154929           23
     summit
83 17739766

R> print(peakTFBS)[18,]

   chrID peakStart peakStop peakSize logAveP  logMinP  aveLogP aveChipCount
18 chr22  18121200 18121799      600 1.36232 4.530337 2.342171           19
   maxChipCount aveInputCount aveInputCountScaled aveLog2Ratio
18           27            11              13.9648     0.373632
```

We can plot the ChIP profile of this H3K4me3 peak with command:

```
R> plot( peakHM, peakNum=8 )
```

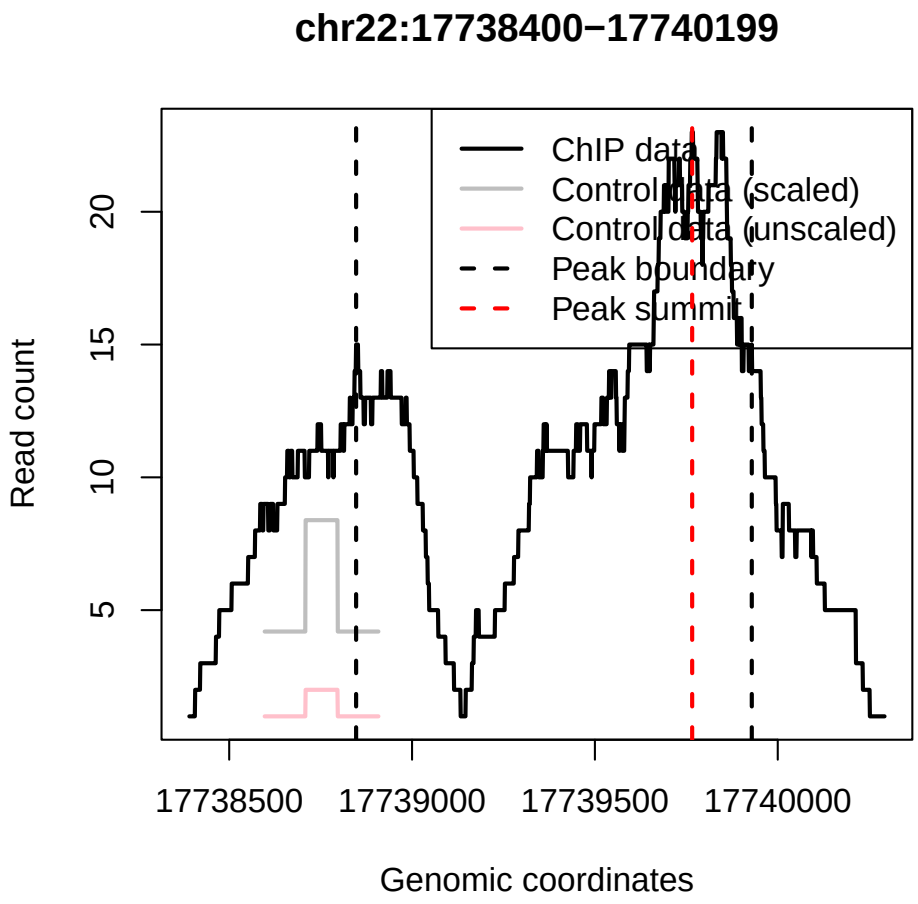


Figure 5: Example ChIP profile plot of H3K4me3 modification in GM12878 cell line.

Figure 5 shows ChIP profile plot of H3K4me3 generated using '`plot`' method. It indicates there is actually strong bimodal ChIP signal in this region.

For more extensive investigation, the wiggle files for STAT1 and H3K4me3 generated using '`generateWig`' method can be used for genome browsers. Figure 6 shows the IGV screenshot of ChIP-seq data of STAT1 and H3K4me3 in the promoter region of BCL2L13, where the 8-th peak of H3K4me3 is located at. It shows that there are overlapping STAT1 binding and H3K4me3 modification in the promoter region of BCL2L13 gene, as suggested by the MOSAiCS and MOSAiCS-HMM peak calling results. This might indicate that BCL2L13 is a target gene of STAT factor and this gene is also transcribed in GM12878 cell line.

6 Two-Sample Analysis using '`mosaicsRunAll`'

Two-sample analysis without mappability and GC content can also be done in a more convenient way, with the command:

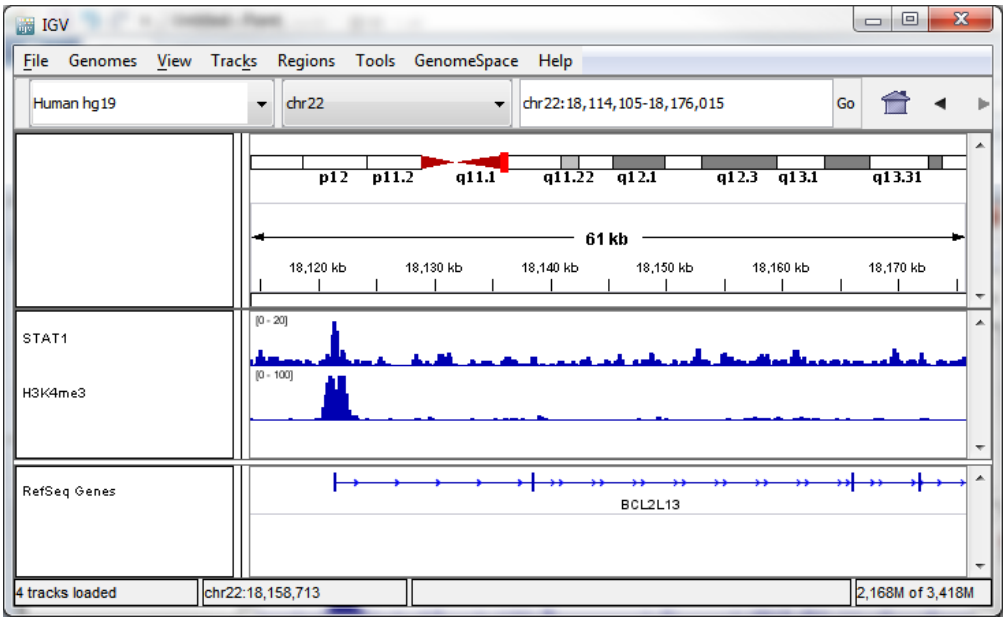


Figure 6: IGV screenshot of ChIP-seq data of STAT1 and H3K4me3 at the promoter region of BCL2L13.

```
R>      mosaicsRunAll(
+          chipFile="/scratch/eland/STAT1_ChIP_eland_results.txt",
+          chipFileFormat="eland_result",
+          controlFile="/scratch/eland/STAT1_control_eland_results.txt",
+          controlFileFormat="eland_result",
+          binfileDir="/scratch/bin/",
+          peakFile=c("/scratch/peak/STAT1_peak_list.bed",
+                     "/scratch/peak/STAT1_peak_list.gff"),
+          peakFileFormat=c("bed", "gff"),
+          reportSummary=TRUE,
+          summaryFile="/scratch/reports/mosaics_summary.txt",
+          reportExploratory=TRUE,
+          exploratoryFile="/scratch/reports/mosaics_exploratory.pdf",
+          reportGOF=TRUE,
+          gofFile="/scratch/reports/mosaics_GOF.pdf",
+          byChr=FALSE, useChrfile=FALSE, chrfile=NULL, excludeChr="chrM",
+          PET=FALSE, FDR=0.05, fragLen=200, binSize=fragLen, capping=0,
+          bgEst="rMOM", signalModel="BIC", parallel=TRUE, nCore=8 )
```

'`mosaicsRunAll`' method imports aligned read files, converts them to bin-level files (generated bin-level files will be saved in the directory specified in '`binfileDir`' argument for future use), fits the MOSAiCS model, identifies peaks, and exports the peak list. In addition, users can also make '`mosaicsRunAll`' method generate diverse analysis reports, such as summary report of parameters and analysis results, exploratory plots, and goodness of fit (GOF) plots. Arguments of '`mosaicsRunAll`' method are summarized in Table 1. See Section 3.1 for details of the arguments '`chipFileFormat`', '`controlFileFormat`', '`byChr`', '`useChrfile`', '`chrfile`', '`excludeChr`', '`PET`', '`fragLen`', '`binSize`', and '`capping`'. See Section 3.3' for details of the argument '`bgEst`'. See Sec-

tion 3.4' for details of the arguments 'FDR', 'signalModel', 'peakFileFormat', 'maxgap', 'minsize', and 'thres'.

7 Two-Sample Analysis with Mappability and GC Content

For the two-sample analysis with mappability and GC content and the one-sample analysis, you also need bin-level mappability, GC content, and sequence ambiguity score files for the reference genome you are working with. If you are working with organisms such as human (HG18 and HG19), mouse (MM9), rat (RN4), and Arabidopsis (TAIR9), you can download their corresponding preprocessed mappability, GC content, and sequence ambiguity score files at http://www.stat.wisc.edu/~keles/Software/mosaics/. If your reference genome of interest is not listed on our website, you can inquire about it at our Google group, http://groups.google.com/group/mosaics_user_group, and we would be happy to add your genome of interest to the list. The companion website also provides all the related scripts and easy-to-follow instructions to prepare these files. Please check http://www.stat.wisc.edu/~keles/Software/mosaics/ for more details.

In this section, we use chromosome 21 data from a ChIP-seq experiment of STAT1 binding in interferon-γ-stimulated HeLa S3 cells [3]. 'mosaicsExample' package also provides this example dataset. You can import bin-level data and fit MOSAiCS model for the two-sample analysis using mappability and GC content with the commands:

```
R> exampleBinData <- readBins( type=c("chip","input","M","GC","N"),
+     fileName=c( system.file( file.path("extdata","chip_chr21.txt"), package="mosaicsExample"),
+     system.file( file.path("extdata","input_chr21.txt"), package="mosaicsExample"),
+     system.file( file.path("extdata","M_chr21.txt"), package="mosaicsExample"),
+     system.file( file.path("extdata","GC_chr21.txt"), package="mosaicsExample"),
+     system.file( file.path("extdata","N_chr21.txt"), package="mosaicsExample") ) )

------------------------------------------------------------
Info: preprocessing summary
------------------------------------------------------------
- percentage of bins with ambiguous sequences: 27%
  (these bins will be excluded from the analysis)
- before preprocessing:
        first coordinates = 0, last coordinates = 46944350
- after preprocessing:
        first coordinates = 9719550, last coordinates = 46944250
------------------------------------------------------------
```

For the 'type' argument, "chip", "input", "M", "GC", and "N" indicate bin-level ChIP data, control sample data, mappability score, GC content score, and sequence ambiguity score, respectively.

When you have mappability and GC contents, 'plot' method provides additional plot types. 'plotType="M"' and 'plotType="GC"' generate plots of mean ChIP tag counts versus mappability and GC content scores, respectively. Moreover, 'plotType="M|input"' and 'plotType="GC|input"' generate plots of mean ChIP tag counts versus mappability and GC content scores, respectively, conditional on control tag counts.

Table 1: **Summary of the arguments of 'mosaicsRunAll' method.**

(a) Input and output files

Argument	Explanation
chipFile	Name of aligned read file of ChIP sample.
chipFileFormat	File format of aligned read file of ChIP sample.
controlFile	Name of aligned read file of matched control sample.
controlFileFormat	File format of aligned read file of matched control sample.
binfileDir	Directory that bin-level files are exported to.
peakFile	Vector of file names of peak list.
peakFileFormat	Vector of file formats of peak list.

(b) Reports

Argument	Explanation
reportSummary *	Generate analysis summary?
summaryFileName	File name of analysis summary.
reportExploratory *	Generate exploratory plots?
exploratoryFileName	File name of exploratory plots.
reportGOF *	Generate GOF plots?
gofFileName	File name of GOF plots.

* Reports will be generated only when these arguments are TRUE. Default is FALSE.

(c) Tuning parameters

Argument	Explanation
byChr	Genome-wide analysis (FALSE) or chromosome-wise analysis (TRUE)?
useChrfile	Use the file containing chromosome ID and chromosome size?
chrfile	Name of the file containing chromosome ID and chromosome size.
excludeChr	Vector of chromosomes to be excluded from the analysis.
fragLen	Average fragment length.
binSize	Bin size.
capping	Cap read counts in aligned read files?
bgEst	Background estimation approach.
signalModel	Signal model.
PET	Paired-end tag (TRUE) or single-end tag (FALSE) ChIP-Seq data?
FDR	False discovery rate (FDR).
maxgap	Distance between initial peaks for merging.
minsize	Minimum width to be called as a peak.
thres	Minimum ChIP tag counts to be called as a peak.
parallel	Use parallel package for parallel computing?
nCore	Number of CPUs used for parallel computing. **

** Relevant only when parallel=TRUE and parallel package is installed.

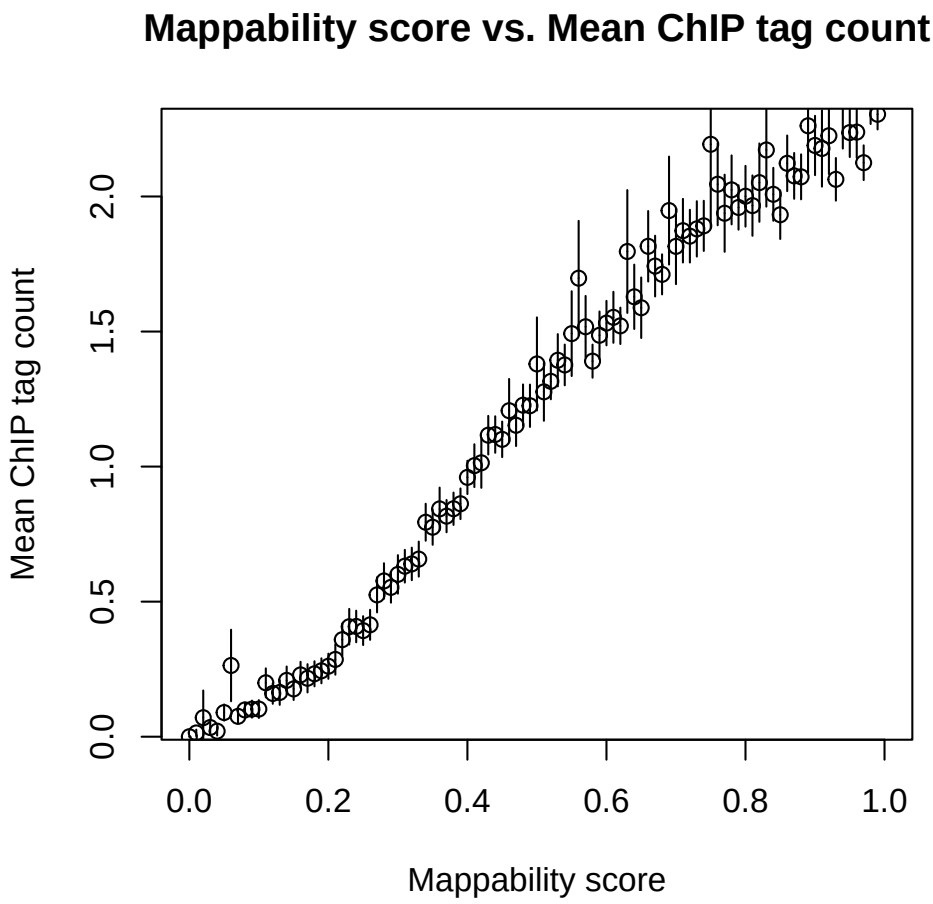


Figure 7: Mean ChIP tag count versus Mappability.

```
R> plot( exampleBinData, plotType="M" )
R> plot( exampleBinData, plotType="GC" )
R> plot( exampleBinData, plotType="M|input" )
R> plot( exampleBinData, plotType="GC|input" )
```

As discussed in [1], we observe that mean ChIP tag count increases as mappability score increases (Figure 7). Mean ChIP tag count depends on GC score in a non-linear fashion (Figure 8). When we condition on control tag counts (Figures 9 and 10), mean ChIP tag count versus mappability and GC content relations exhibit similar patterns to that of marginal plots given in Figures 7 and 8. MOSAiCS incorporates this observation by modeling ChIP tag counts from non-peak regions with a small number of control tag counts as a function of mappability, GC content, and control tag counts.

Application of MOSAiCS to multiple case studies of ChIP-seq data with low sequencing depth showed that consideration of mappability and GC content in the model improves sensitivity and specificity of peak identification even in the presence of a control sample [1]. `mosaics` package accommodates a two-sample analysis with mappability and GC content by specification of

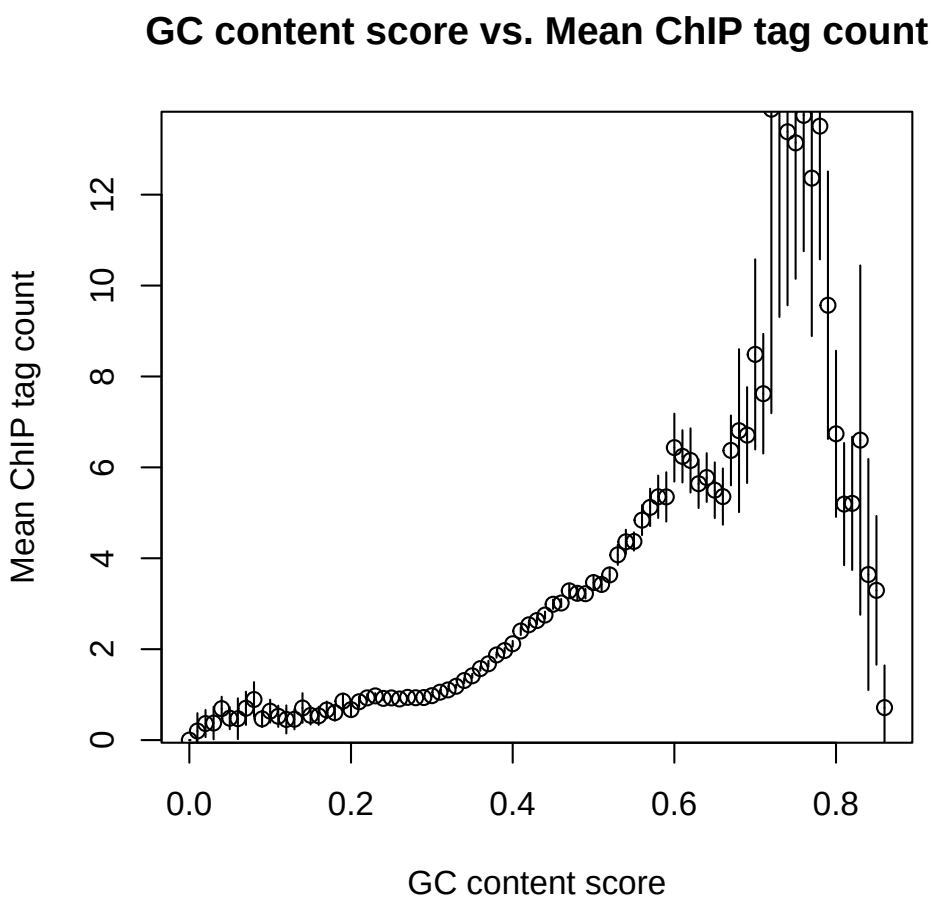


Figure 8: Mean ChIP tag count versus GC content.

'`analysisType="TS"`' when calling the '`mosaicsFit`' method.

```
R> exampleFit <- mosaicsFit( exampleBinData, analysisType="TS", bgEst="automatic" )
```

Peak identification can be done exactly in the same way as in the case of the two-sample analysis without mappability and GC content.

```
R> OneSamplePeak <- mosaicsPeak( OneSampleFit, signalModel="2S", FDR=0.05,
+ maxgap=200, minsize=50, thres=10 )
```

8 One-Sample Analysis

When control sample is not available, '`mosaics`' package accommodates one-sample analysis of ChIP-seq data. Implementation of the MOSAiCS one-sample model is very similar to that of the two-sample analysis. Bin-level data for the one-sample analysis can be imported to the R environment with the command:

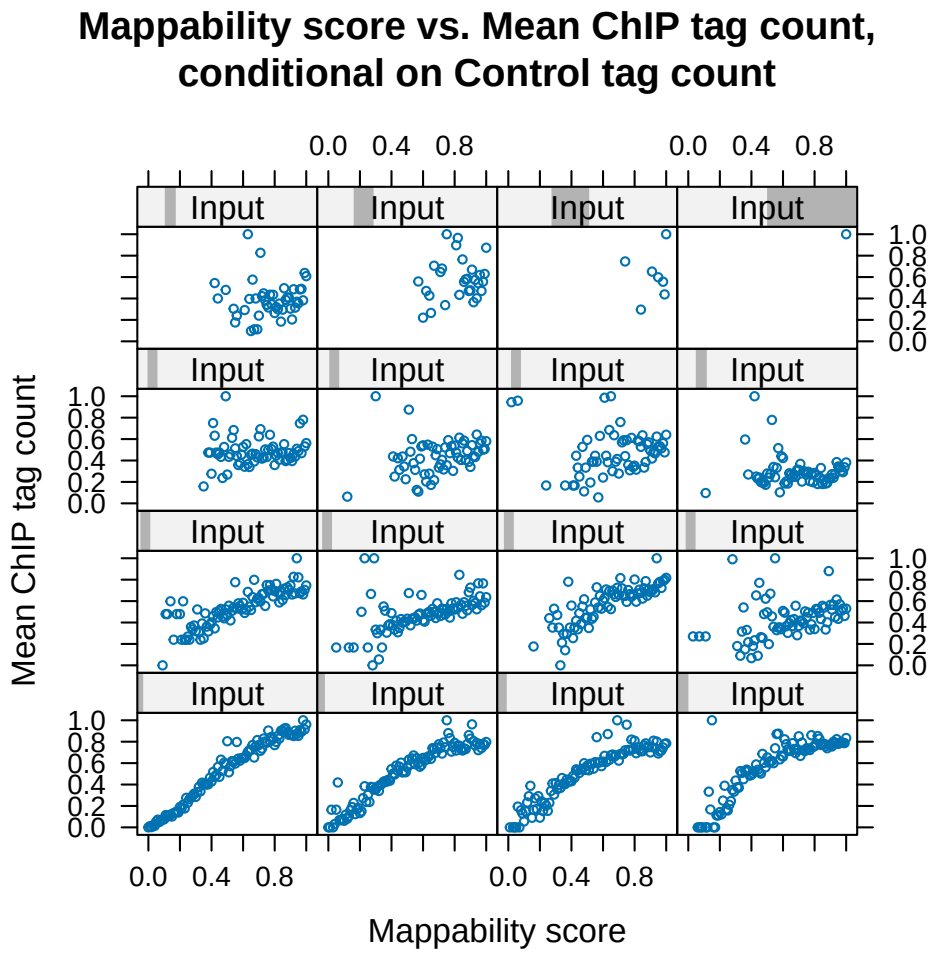


Figure 9: Mean ChIP tag count versus Mappability, conditional on control tag counts.

```
R> exampleBinData <- readBins( type=c("chip","M","GC","N"),
+     fileName=c( system.file( file.path("extdata","chip_chr21.txt"), package="mosaicsExample"),
+     system.file( file.path("extdata","M_chr21.txt"), package="mosaicsExample"),
+     system.file( file.path("extdata","GC_chr21.txt"), package="mosaicsExample"),
+     system.file( file.path("extdata","N_chr21.txt"), package="mosaicsExample") ) )
```

In order to fit a MOSAiCS model for the one-sample analysis, you need to specify '`analysisType="OS"`' when calling the '`mosaicsFit`' method.

```
R> exampleFit <- mosaicsFit( exampleBinData, analysisType="OS", bgEst="automatic" )
```

Peak identification can be done exactly in the same way as in the case of the two-sample analysis.

```
R> examplePeak <- mosaicsPeak( exampleFit, signalModel="2S", FDR=0.05,
+ maxgap=200, minsize=50, thres=10 )
```

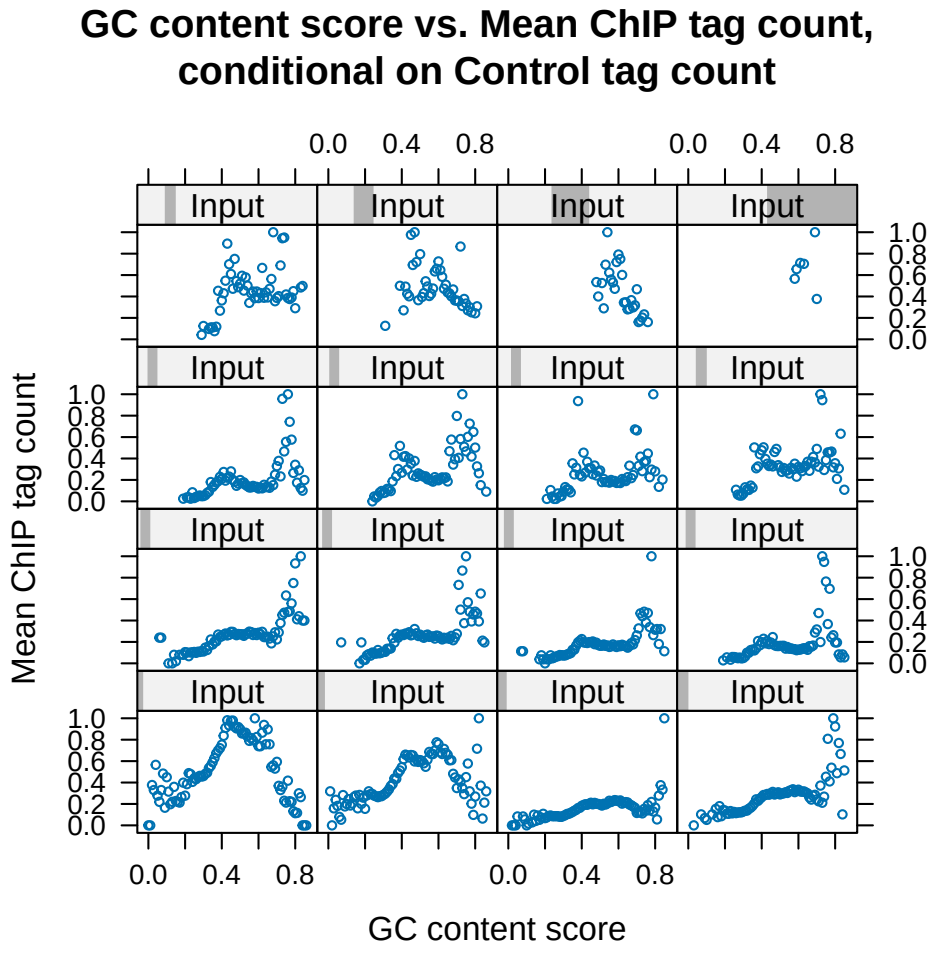


Figure 10: Mean ChIP tag count versus GC content, conditional on control tag counts.

9 Case Studies: Tuning Parameters to Improve the MOSAiCS Fit

Because the `mosaics` package is based on the statistical modeling approach, it is important to check that we have nice model fits. Goodness of fit (GOF) plots generated from the '`plot`' method are key tools that users can utilize to check the MOSAiCS fit and improve it if necessary. In this section, we will discuss several case studies based on real ChIP-Seq data, especially focusing on understanding unsatisfactory MOSAiCS fits and suggesting strategies to improve it. The case studies illustrated in this section are based on [4] and we provided deeper discussion and example analysis codes in this book chapter.

9.1 Case 1

Figure 11a shows the GOF plot for the two-sample analysis without using mappability and GC content. The GOF plot indicates that ChIP tag counts simulated from the fitted model are on average higher than the actual ChIP-Seq data. Moreover, estimated background dominates in the fitted model. In such cases (*over-estimation of background*), we usually have much smaller number

of peaks. This problem occurred essentially because '`mosaicsFit`' guessed using bins with low ChIP tag counts ('`bgEst="matchLow"`') as the optimal background estimation approach, from its automatic selection ('`bgEst="automatic"`'). Although automatic selection usually works well, there are some cases that it does not result in optimal background estimation approach. In this case, the MOSAiCS fit can be improved by explicitly specifying the background estimation approach with the command:

```
R> exampleFit <- mosaicsFit( exampleBinData, analysisType="IO", bgEst="rMOM" )
```

Figure 11b shows that the MOSAiCS fit was significantly improved by explicitly using the robust method of moment approach ('`bgEst="rMOM"`').

9.2 Case 2

Figure 12a shows the GOF plot for the two-sample analysis without using mappability and GC content. The GOF plot indicates that the MOSAiCS fit is unsatisfactory for the bins with low ChIP tag counts. This problem occurred essentially because background estimation was affected by some bins with high tag counts in the matched control data. In the two-sample analysis without using mappability and GC content, the '`truncProb`' argument indicates proportion of bins that are not considered as outliers in the control data and it controls the functional form used for the control data. Although our empirical studies indicates that default '`truncProb`' value usually works well, there are some cases that it does not result in optimal background estimation. In this case, the MOSAiCS fit can be improved by lowering the '`truncProb`' value with the command:

```
R> exampleFit <- mosaicsFit( exampleBinData, analysisType="IO",
+         bgEst="automatic", truncProb=0.80 )
```

Figure 12b shows that the MOSAiCS fit was significantly improved by using '`truncProb = 0.80`'.

9.3 Case 3

Figure 13a displays the GOF plot for the two-sample analysis without utilizing mappability and GC content. Neither the blue (two-signal component) nor the red (one-signal component) curve provides good fit to the ChIP data. In this case, we consider fitting a two-sample analysis with mappability and GC content. Figure 13b displays the GOF plot for the two-sample analysis with mappability and GC content in addition to Input and the goodness of fit improves significantly by utilizing mappability and GC content.

9.4 Note on Parameter Tuning

Overall, unsatisfactory model fits can be improved via the tuning parameters in the '`mosaicsFit`' function. Our empirical studies suggest that as the sequencing depths are getting larger, genomic features mappability and GC content have less of an impact on the overall model fit. In particular, we suggest tuning the the two-sample model without mappability and GC content in the cases of unsatisfactory fits before switching to a fit with mappability and GC content. The following are some general tuning suggestions: for the two-sample analysis without utilizing mappability and GC content, lowering the value of the '`truncProb`' parameter; for the two-sample analysis with mappability and GC content, a larger '`s`' parameter and a smaller '`meanThres`' parameter; for the one-sample analysis with mappability and GC content, varying the '`meanThres`' parameter are the

general tuning guidelines. Although MOSAiCS has two additional tunable parameters, '`k`' and '`d`', we have accumulated ample empirical evidence through analyzing many datasets that the default values of '`k = 3`' and '`d = 0.25`' work well for these parameters [4]. If you encounter a fitting problem you need help with, feel free to contact us at our Google group, `http://groups.google.com/group/mosaics_user_group`.

10 Conclusion and Ongoing Work

R package `mosaics` provides effective tools to read and investigate ChIP-seq data, fit MOSAiCS model, and identify peaks. We are continuously working on improving `mosaics` package further, especially in supporting more diverse genomes, automating fitting procedures, developing more friendly and easy-to-use user interface, and providing more effective data investigation tools. Please post any questions or requests regarding '`mosaics`' package at `http://groups.google.com/group/mosaics_user_group`. Updates and changes of '`mosaics`' package will be announced at our Google group and the companion website (`http://www.stat.wisc.edu/~keles/Software/mosaics/`).

Acknowledgements

We thank Gasch, Svaren, Chang, Kiley, Bresnick, Pike, and Donohue Labs at the University of Wisconsin-Madison for sharing their data for MOSAiCS analysis and useful discussions. We also thank Colin Dewey and Bo Li for the CSEM output in Appendix A.7.

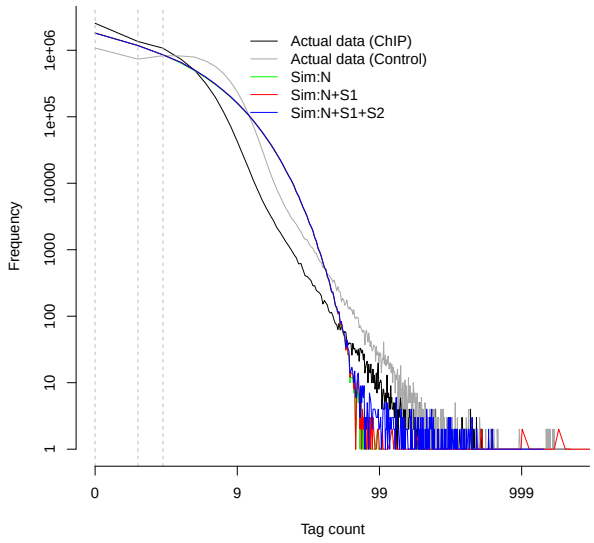
References

[1] Kuan, PF, D Chung, G Pan, JA Thomson, R Stewart, and S Keleş (2010), "A Statistical Framework for the Analysis of ChIP-Seq Data", *Journal of the American Statistical Association*, 106, 891-903.

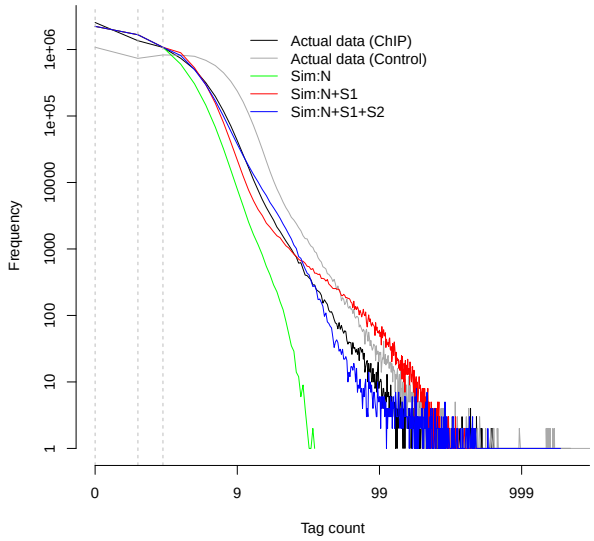
[2] Chung, D, Zhang Q, and Keleş S (2014), "MOSAiCS-HMM: A model-based approach for detecting regions of histone modifications from ChIP-seq data", Datta S and Nettleton D (eds.), *Statistical Analysis of Next Generation Sequencing Data*, Springer.

[3] Rozowsky, J, G Euskirchen, R Auerbach, D Zhang, T Gibson, R Bjornson, N Carriero, M Snyder, and M Gerstein (2009), "PeakSeq enables systematic scoring of ChIP-Seq experiments relative to controls", *Nature Biotechnology*, 27, 66-75.

[4] Sun, G, D Chung, K Liang, S Keleş (2012), "Statistical Analysis of ChIP-Seq Data with MOSAiCS", *Methods in Molecular Biology*, 1038, 193-212.

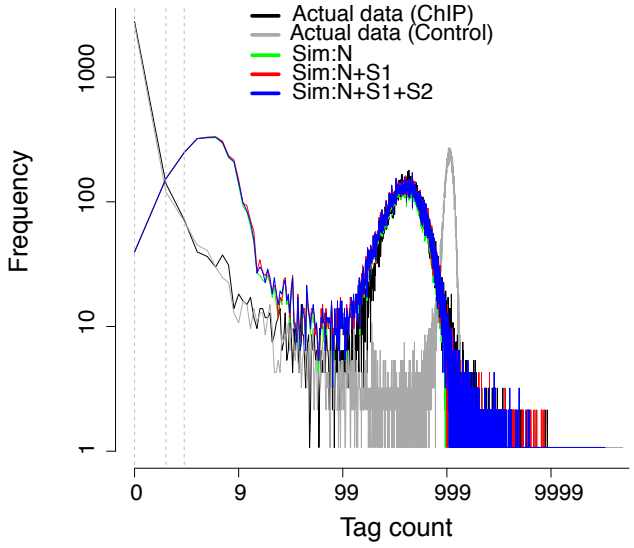


(a) When the background estimation approach was automatically chosen

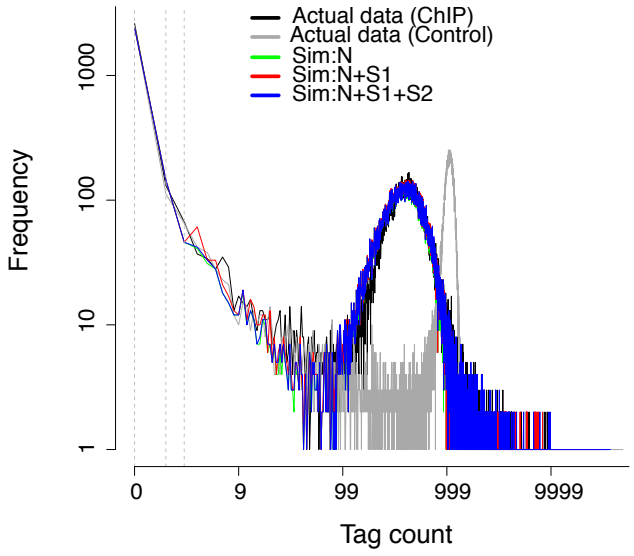


(b) When the background estimation approach was explicitly specified

Figure 11: Case #1. Goodness of fit when the background estimation approach was automatically chosen by `mosaics` package (a) and explicitly specified as robust method of moment (b). The MOSAiCS fit was significantly improved by explicitly specifying the background estimation approach.

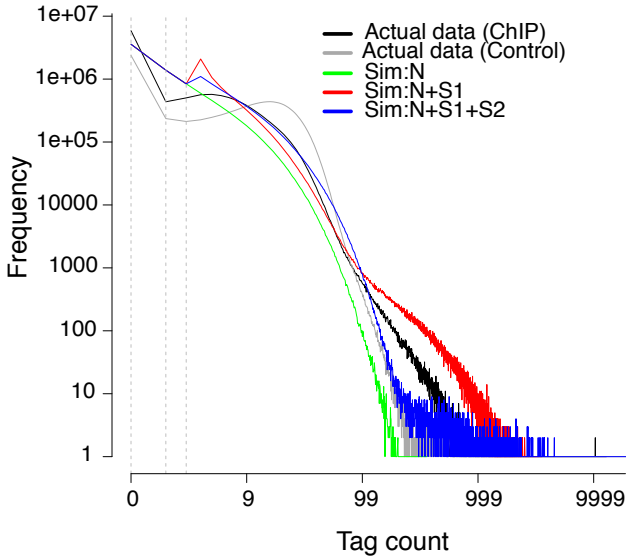


(a) '`truncProb = 0.999`' (default)

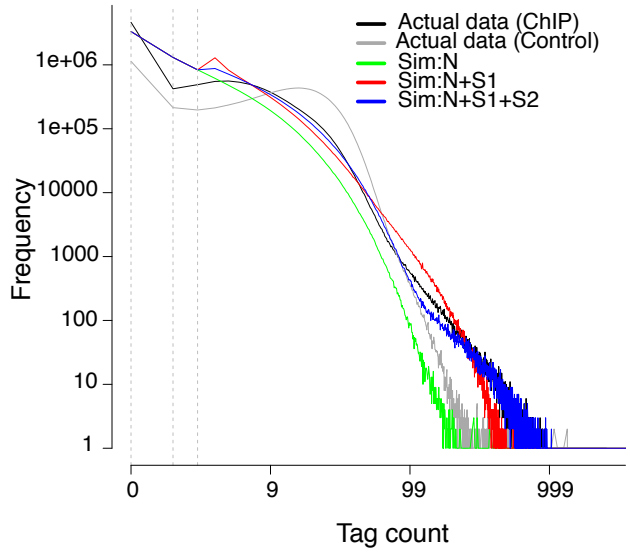


(b) '`truncProb = 0.80`'

Figure 12: Case #2. Goodness of fit when default '`truncProb`' value (0.999) is used (a) and '`truncProb = 0.80`' is explicitly specified (b). The MOSAiCS fit was significantly improved by lowering the '`truncProb`' value.



(a) Two-sample analysis without mappability and GC content



(b) Two-sample analysis with mappability and GC content

Figure 13: Case #3. Goodness of fit for the two-sample analysis without utilizing mappability and GC content (a). The MOSAiCS fit was improved when we use the two-sample analysis utilizing mappability and GC content (b).

A Appendix: Example Lines of Aligned Read Files for SET ChIP-Seq Data

A.1 Eland Result File Format

```
>HWUSI-EAS787_0001:6:1:2020:1312#NTAGGC/1 TTGCGGGTTAACAAAAACGATGTAAACCATCG U0 1 0 0 U00096
3894480 F ..
>HWUSI-EAS787_0001:6:1:2960:1312#NTAGGC/1 TATTTTCCTGGTTCTGCTGCTGGTGAGCGGCG U0 1 0 0 U00096
4079496 R ..
>HWUSI-EAS787_0001:6:1:3045:1310#NTAGGC/1 AGCGTATCAAAACTGACAATTCATTCTATGAA U0 1 0 0 U00096
2087194 F ..
>HWUSI-EAS787_0001:6:1:3887:1310#NTAGGC/1 TCCGAGTTGTCGCAATGGGGGCAAGGTGTGAA U1 0 1 0 U00096
2405216 R .. 26C
>HWUSI-EAS787_0001:6:1:3993:1315#TTAGGC/1 AGACATAGTAAAACCGATACCGCACAGGATCC U0 1 0 0 U00096
18483 R ..
>HWUSI-EAS787_0001:6:1:4156:1312#NTAGGC/1 GCCGAACTGACAAATAAAATAGCCTGATAGGA U0 1 0 0 U00096
2687648 F ..
>HWUSI-EAS787_0001:6:1:4215:1310#NTAGGC/1 GGAATTTCACGAAAACAGAGCTAAAGCGCCGT U0 1 0 0 U00096
4569776 F ..
>HWUSI-EAS787_0001:6:1:4458:1311#NTAGGC/1 GTTCCTGGCAGGGATTCGGCGCACTCGCCCAC U0 1 0 0 U00096
3607517 F ..
>HWUSI-EAS787_0001:6:1:4841:1308#NTAGGC/1 GGCGTGTGCATCAGCAGAATTTTTCGGGCGGT U0 1 0 0 U00096
3715588 R ..
>HWUSI-EAS787_0001:6:1:5215:1316#TTAGGC/1 TGATATTCTCCGCAGCCAGACTTTTTCCGCCA U0 1 0 0 U00096
668432 R ..
```

A.2 Eland Extended File Format

```
>SOLEXA2_1110:7:1:1599:1010#0/1 NNTTTATTTCAACAGGATACTAATATCCTAGGCCCTTTTC 1:0:0 chr21.fa:26418321
>SOLEXA2_1110:7:1:2734:1009#0/1 NNCCCCTCCAATGACTATAGCAATGATCTTCATGGTAGAT 1:0:0 chr18.fa:37353206
>SOLEXA2_1110:7:1:3130:1007#0/1 NNTCTTATATTCTTACGAAGCATCAGTCTTAGCAGAAACT 1:0:0 chr4.fa:141242065
>SOLEXA2_1110:7:1:3709:1009#0/1 NNAAAAAATACAAAAATTAGGCTGTGCACGGTGGTGCAGG 0:1:0 chr18.fa:12860297
>SOLEXA2_1110:7:1:4887:1009#0/1 NNTTATATAACAATTCAACAACTATTTCTTGAGCATCTAA 1:0:0 chr13.fa:60908544
>SOLEXA2_1110:7:1:6298:1011#0/1 NNAGACACACATTTGTTTTTTGTATTTTTCACATCTCTTT 1:0:0 chr11.fa:47506195
>SOLEXA2_1110:7:1:7798:1011#0/1 NNAGACACTGAGATTTTATGATACTGATTATTGTCGCATT 1:0:0 chr11.fa:25291690
>SOLEXA2_1110:7:1:7970:1007#0/1 NNGGCTCCTAAAAAGCATACTTTTTTTTTTGTTTTCTGTAT 0:0:1 chr21.fa:19842628
>SOLEXA2_1110:7:1:8051:1007#0/1 NNTCACATTACCTTTACTAATCCTGAGAGAGCAGGAACTC 1:0:0 chr7.fa:47191653F
>SOLEXA2_1110:7:1:8422:1011#0/1 NNGGGCTTTTGGGGATGATGGTGTCACCAGGTGGGGAATT 1:0:0 chr1.fa:59480447R
```

A.3 Eland Export File Format

```
AMELIA 102 5 1 7564 6792 ...T.. 1 TCTCATTTTTTACGTTTTTCAGTGATTTCGTCATTTTTCAAGTCGTCAAG
ggggggggggggggggggggggggeggggggggeggggggeeggggggggggg 0:48:71 Y
AMELIA 102 5 1 7589 6796 ...A.. 1 TGTGCATTTCACATTTTTCACGTTTTTTAGTGATTTCGTCATTTTTCAAG
gggggggggggggggggggggggggggggfgeggggggggggggggggggg chr2.fa 98506985 R 50 40 Y
AMELIA 102 5 1 7721 6804 ...G.. 1 TATGAGACAGTGGATATGTTATGTAATATCTCCAAGTCAACTTCTTTTCA
cca`caabbceceeaca\ccbccZc[[_\^a_\cbdcdc_YWRPXa^BBB chrX.fa 18454546 F 50 190 Y
AMELIA 102 5 1 7516 6810 ...C.. 1 GGGACAGTTCTGCAGACTTGGGACTCCACTCTAGAGAGGAAGAAAGAGAG
RUQKUV[\]S_BBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBB chr1.fa 101923817 F 25A12A11 29 N
```

```
AMELIA 102 5 1 7650 6808 ...C.. 1 AGAGGTGCCTGCACCTCCAGCCACCTGGGAGGCTGTGTCAGGAGGATCAC
dddddaeffegggggegggfegggggggggabcdegg^eged`bdb]_cb chr12.fa 80471182 F 50 203 Y
AMELIA 102 5 1 7703 6811 ...C.. 1 CTTTTAGATAACAATCCCATATTCAGGAAGTTTTATTCAATTCATTCAAG
gggfeggggffffgggggfgfgdgffgggfddfeddfff^gfgagggagg chr4.fa 17871000 F 50 203 Y
AMELIA 102 5 1 7524 6826 ...A.. 1 ACTTCGCCTTACTGTAAGTGTATCTCACGCATGCAGAGCTCAGCAGCGGT
BBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBB NM N
AMELIA 102 5 1 7600 6825 ...T.. 1 GATATATAAATATTTATTATATGTAAAACACGTATTTTAAAGAACTTAAA
gggggfggfggfgffgfggfddeddgggdgfdaeeggaeeffdgegbgge chr15.fa 57197391 R 50 203 Y
AMELIA 102 5 1 7564 6837 ...A.. 1 TTCGGCTTGGCAGCAAGCGCCAACTACCCCCTAAGCCAGCTTTCGAGCCT
bbbbbbbbK^BBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBB chr1.fa 129403054 R 29A20 27 N
AMELIA 102 5 1 7620 6840 ...A.. 1 CTTGACGACTTGAAAAATGACGAATTCACTAAAATACGTGAAAAAAGAGA
fggffgfdggeeffdfdffdedfgggfeeebddddfdfebbdd_dgggge 0:4:13 Y
```

A.4 Default Bowtie File Format

```
HWI-EAS255_8232_FC30D82_6_1_11_1558 -chr7 82587091 ATCTTGTTTTATCTTGAAAAGCAGCTTCCTTAAAC
IIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIII 0
HWI-EAS255_8232_FC30D82_6_1_12_793 + chr4 33256683 GTTAATCGGGAAAAAAACCTTTTCCAGGAAAACAA
IIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIII 0
HWI-EAS255_8232_FC30D82_6_1_17_606 -chr12 92751594 GCCTACATCCTCAGATTCATACATGTCACCATTTC
IIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIII 0
HWI-EAS255_8232_FC30D82_6_1_29_1543 + chr20 53004497 GTGACCTAGTTAAATACTGGCTCCACCTCTTGGAG
IIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIII 0
HWI-EAS255_8232_FC30D82_6_1_22_327 + chr8 93861371 GAAATAGCAGGAGACAGGTACTTTATCTTGCTTTT
IIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIII 0
HWI-EAS255_8232_FC30D82_6_1_10_1544 -chr4 121630802 TTCCCTAACATCTGTGTCAATTCTCTGTCAACTGC
IIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIII 0
HWI-EAS255_8232_FC30D82_6_1_108_1797 + chr8 131141865 GTTGAATGAAATGGATATGAAACAAAGCACTATAC
IIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIII 0
HWI-EAS255_8232_FC30D82_6_1_41_1581 -chr7 16554954 CTCTTGCTCTCTTCATTAGTTTAGTTTTCTTCTAC
IIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIII 0 22:G>T
HWI-EAS255_8232_FC30D82_6_1_12_787 -chr8 119427380 CCCTAGAGGAGCTCAAAACAACTGCACAACACAAC
IIIEIIIIIIIIIIIIIIIIIIIIIIIIIIIIIII 0 23:T>C
HWI-EAS255_8232_FC30D82_6_1_6_1497 + chr1 233856805 GTTAAAGGCGTTTGGATATGATGCAATTCCAAATC
IIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIII 0
```

A.5 SAM File Format

```
SALLY:187:B02JLABXX:5:1101:1418:1923 65 U00096 1943409 30 51M * 0 0 GTTCGCGAATTAAAGTTTCACTGCTGGC
#1:DB??@F?DFFB@<E?CGCHIICHG<G?FHOB?76@FG=8?;;>DA@C>NM:i:1 MD:Z:0C50
SALLY:187:B02JLABXX:5:1101:1441:1969 65 U00096 3310343 30 51M * 0 0 GCGGACGCGCTTCACGCGGAACTTCAAT
#1=BDBDDHA?D?ECCGDE:7@BGCEI@CGGGICGHE6BABBBCDCC>A?@ NM:i:1 MD:Z:0T50
SALLY:187:B02JLABXX:5:1101:1685:1925 65 U00096 2164008 30 51M * 0 0 GGCACTTCCCTGAAATGCCGATCCACCT
#1=DFFFFHHHGHIIIIIIIEHGIIIIIIIIIFCHIIIIIHIGGGHH@BH NM:i:1 MD:Z:0A50
SALLY:187:B02JLABXX:5:1101:1587:1933 65 U00096 2140205 30 51M * 0 0 GGCCAGGCTTCAATATCTCGGTCACACA
#1=DDFFFHHHHGIJIIJJJJHIGIJJJJJJIJJJJJHIJJJJJJJJJJJJ NM:i:1 MD:Z:0A50
SALLY:187:B02JLABXX:5:1101:1635:1986 81 U00096 1432777 30 51M * 0 0 AACGTGCTGCGGTTGGTTTGACTTTGAT
HEJJJJJJJJIJJGJJJJJIHJJJJJJIGJJIHJJJJJHHHHHDFFFD=1# NM:i:0 MD:Z:51
```

```
SALLY:187:B02JLABXX:5:1101:1607:2000 65 U00096 932738 30 51M * 0 0 GGGCGTCATCAGTCCAGCGACGAATACAT
#1=BDD?DFFFFFI<AE>GIFFBGFIIFGIBFIIFIEIIIFEF;ADFFAEI NM:i:1 MD:Z:0T50
SALLY:187:B02JLABXX:5:1101:1937:1904 81 U00096 4541497 30 51M * 0 0 AAACGTTGGTGTGCAGATCCTGGACAGA
JJIJJJJJJJJJJJIJJJJJIHHIGIIJJJJJIGIJJJHHHHHFFFDD=4# NM:i:1 MD:Z:50A0
SALLY:187:B02JLABXX:5:1101:1881:1942 65 U00096 4003009 30 51M * 0 0 GGCTGCAGGAGCATGACAATCCGTTCAC
#1:BDFA>FDHBHHIGHICFFGGHE3??FFEHGCCBFFC@FHIGFFFFIII NM:i:1 MD:Z:0T50
SALLY:187:B02JLABXX:5:1101:1919:1960 81 U00096 2237700 30 51M * 0 0 GCGTTCGGGCCAGACAGCCAGGCGTCCA
###############B(;?BGGBFGBD??99?9CBAE@AFFDFD1D?=11# NM:i:1 MD:Z:50G0
SALLY:187:B02JLABXX:5:1101:1900:1973 65 U00096 442575 30 51M * 0 0 GCGGTGGAACTGTTTAACGGTTTCAACCA
#1=DBDFFHHGHHHIJJJJJGHIHJJHIII=FGAGBFGGIJJJJJEIJJGG NM:i:1 MD:Z:0A50
```

A.6 BED File Format

```
chrA 880 911 U 0 +
chrA 883 914 U 0 +
chrA 922 953 U 0 +
chrA 931 962 U 0 +
chrA 950 981 U 0 +
chrA 951 982 U 0 +
chrA 959 990 U 0 +
chrA 963 994 U 0 +
chrA 965 996 U 0 +
chrA 745 776 U 0 -
```

A.7 CSEM File Format

```
chr11 114728597 114728633 HWUSI-EAS610:1:1:0:647#0/1 1000.00 +
chr5 138784256 138784292 HWUSI-EAS610:1:1:0:498#0/1 1000.00 -
chr3 8516078 8516114 HWUSI-EAS610:1:1:0:631#0/1 1000.00 -
chr7 123370863 123370899 HWUSI-EAS610:1:1:0:508#0/1 1000.00 +
chr4 137837148 137837184 HWUSI-EAS610:1:1:0:1790#0/1 1000.00 -
chr11 84363281 84363317 HWUSI-EAS610:1:1:0:862#0/1 1000.00 -
chr7 66950830 66950866 HWUSI-EAS610:1:1:0:1675#0/1 371.61 -
chr7 66938672 66938708 HWUSI-EAS610:1:1:0:1675#0/1 628.39 -
chr15 57549345 57549381 HWUSI-EAS610:1:1:0:969#0/1 1000.00 -
chr9 3012912 3012948 HWUSI-EAS610:1:1:0:194#0/1 448.96 +
```

B Appendix: Example Lines of Aligned Read Files for PET ChIP-Seq Data

B.1 Eland Result File Format

```
>SALLY:194:B045RABXX:3:1101:1225:2090 GAGCAACGGCCCGAAGGGCGAGACGAAGTCGAGTCA U0 1 0 0 U00096
779900 R ..
>SALLY:194:B045RABXX:3:1101:1225:2090 GTTGCCACGGGATATCAAACAAACCGAAAGCAACGA U0 1 0 0 U00096
779735 F ..
>SALLY:194:B045RABXX:3:1101:1197:2174 GGTTGTCTTCCGGGTTGAGGCGCGGAATGTCGAACG   U0 1 0 0
U00096 4080317 R ..
>SALLY:194:B045RABXX:3:1101:1197:2174 GTTTTCAGCTCAGCAACGCGCTCGCTCGCCAGCGTT   U0 1 0 0
U00096 4080169 F ..
>SALLY:194:B045RABXX:3:1101:1448:2081 TGAAATTCATCATGGCTGATACTGGCTTTGGTAAAA   U0 1 0 0
U00096 2474369 F ..
>SALLY:194:B045RABXX:3:1101:1448:2081 CCAATGGGTAAAATAGCGGGTAAAATATTTCTCACA   U0 1 0 0
U00096 2474522 R ..
>SALLY:194:B045RABXX:3:1101:1460:2102 ACATGTTCTTATTCTTACCTCGTAATATTTAACGCT   U0 1 0 0
U00096 2127671 R ..
>SALLY:194:B045RABXX:3:1101:1460:2102 TTTCCAGATACTCACGGGTGCCGTCGTTGGAACCGC   U0 1 0 0
U00096 2127518 F ..
>SALLY:194:B045RABXX:3:1101:1319:2153 AATGAAATGCTGTTTTCATAAAAAATAAAATTGAAG   U0 1 0 0
U00096 1785339 R ..
>SALLY:194:B045RABXX:3:1101:1319:2153 TTATCTTTGTATATTTAACCGGATGAAAAAAACGGT   U1 0 1 0
U00096 1785190 F ..
```

B.2 SAM File Format

```
AMELIA:216:COLAWACXX:4:1101:1138:2071 99 U00096.2 182616 255 65M = 182749 198 TTGCTGGTACTTTCACAG
@@CFFFFDHFFFHBG@FAFEGGGGHHIIIBHIGCHHIHGIIIIIHIHHDDD@A>;?BDDDCCDDD XA:i:0 MD:Z:65 NM:i:0
AMELIA:216:COLAWACXX:4:1101:1138:2071 147 U00096.2 182749 255 65M = 182616 -198 GTGAACCAGAGAATCT
HIIIGJIJJJIJIHGGGFGCIJGIIJJIHDIHGJJJJJIIJJIIHIFJIIJJHHHHHFDFFFC@C XA:i:0 MD:Z:65 NM:i:0
AMELIA:216:COLAWACXX:4:1101:1425:2097 99 U00096.2 4381249 255 65M = 4381349 165 TGTTTACCTTTGGCGT
CCBFFFFFHHHHHJJIJJJJJJJJJJJJJJJJJJJJJJJJIJIJJJJJJJIIIIJJGIHHHHGFF XA:i:0 MD:Z:65 NM:i:0
AMELIA:216:COLAWACXX:4:1101:1425:2097 147 U00096.2 4381349 255 65M = 4381249 -165 AGATCATCGGGTCG
FFFFFFHHJJJJJIHIJJIJIJIEIJJJIJIJJJJJIJJJJJJJJIIJJIJHHHHHFFFFFCCC XA:i:0 MD:Z:65 NM:i:0
AMELIA:216:COLAWACXX:4:1101:1213:2207 163 U00096.2 3632282 255 65M = 3632429 212 CCATGTTGCAGGAAA
CCCFFFFFHHHHHJJJJJJJIIGJJJJIJJJJJGIJIGIGIJJJJJJJJJIJJJJJJJJIJGIJJ XA:i:0 MD:Z:65 NM:i:0
AMELIA:216:COLAWACXX:4:1101:1213:2207 83 U00096.2 3632429 255 65M = 3632282 -212 TTATCTTGATGGCTT
HJIGIJIJJIGEJIIIJJIJGJJIIIJJIJJJIIHIGJJJJJJJIIJIJJJIHHHHHEDFFFCCB XA:i:0 MD:Z:65 NM:i:0
AMELIA:216:COLAWACXX:4:1101:1167:2244 163 U00096.2 4135404 255 65M = 4135541 202 CCAGTTTCTGCAAAT
CCCFFFFFHHHHHJJJIJJJJJIJJJJJJJJJJJIJJJJJJJHGHGFFFFFDDCDDDDDDDDD XA:i:0 MD:Z:65 NM:i:0
```

C Appendix: Chromosome Information File

The first and second columns indicate chromosome ID and its size, respectively.

```
chr1    249250621
chr2    243199373
chr3    198022430
chr4    191154276
chr5    180915260
chr6    171115067
chr7    159138663
chr8    146364022
chr9    141213431
chr10   135534747
```