Analysis of ChIP-seq Data with '`mosaics`' Package

Dongjun Chung[1], Pei Fen Kuan[2] and Sündüz Keleş[1,3]
[1]Department of Statistics, University of Wisconsin
Madison, WI 53706.
[2]Department of Biostatistics, University of North Carolina at Chapel Hill
Chapel Hill, NC 27599.
[3]Department of Biostatistics and Medical Informatics, University of Wisconsin
Madison, WI 53706.

February 8, 2012

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 control samples) ChIP-seq data. It accounts for mappability and GC content biases that arise in ChIP-seq data.

The package can be loaded with the command:

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

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 (either with mappability and GC content or without them) is usually recommended. The two-sample analysis with mappability and GC content is especially useful for the ChIP-seq data with low sequencing depth. If the ChIP-seq data is deeply sequenced, the two-sample analysis is often appropriate. 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. Sections 3 and 4 discuss the work flow and command lines for the two-sample analysis with mappability and GC content and without them, respectively. Section 5 discuss the case of the one-sample analysis.

'`mosaics`' package accepts aligned read files (such as ones obtained from the ELAND or bowtie aligner) as input and converts them into bin-level files for modeling and visualization purposes. 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. We encourage questions or requests regarding 'mosaics' package to be posted on our Google group http://groups.google.com/group/mosaics_user_group.

3 Workflow: Two-Sample Analysis

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. These bin-level data can easily be generated from the aligned read files with the command:

```
R> constructBins( infileLoc="/scratch/eland/", infileName="STAT1_eland_results.txt",
+     fileFormat="eland_result", outfileLoc=infileLoc,
+     byChr=FALSE, fragLen=200, binSize=fragLen, capping=0 )
```

You can specify the directory, name, and file format of the aligned read file in 'infileLoc', 'infileName', and 'fileFormat' arguments, respectively. 'constructBins' method currently allows the following aligned read file formats: Eland result ('"eland_result"'), Eland extended ('"eland_extended"'), Eland export ('"eland_export"'), default Bowtie ('"bowtie"'), SAM ('"sam"'), BED ('"bed"'), and CSEM BED ('"csem"'). This method assumes that these aligned read files are obtained from single-end tag (SET) experiments. If input file format is neither BED nor CSEM BED, it retains only reads mapping uniquely to the reference genome (uni-reads).

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 available in Galaxy Tool Shed (http://toolshed.g2.bx.psu.edu/; "csem" under "Next Gen Mappers"). Stand-alone version of CSEM is also available at http://www.stat.wisc.edu/~keles/Software/multi-reads/. 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"'.

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. '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'.

'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]_bin[binSize].txt', 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 '[chrID]_[infileName]_fragL[fragLen] where [chrID] is chromosome ID that reads align to. These chromosome IDs ([chrID]) are extracted from the aligned read file. By default, constructed bin-level files are exported to the directory that the aligned read file is located at. If you prefer to export them to another directory, you can specify it in 'outfileLoc' argument.

3.2 Reading Bin-Level Data into the R Environment

For the two-sample analysis with mappability and GC content, you need preprocessed bin-level ChIP data, control sample data, mappability score, GC content score, and sequence ambiguity score. In this vignette, we use chromosome 21 data from a ChIP-seq experiment of STAT1 binding in interferon-γ-stimulated HeLa S3 cells [2]. 'mosaicsExample' package provides this example dataset.

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

Bin-level data can be imported to the R environment with the command:

```
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. 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 and it is recommended for most cases. The chromosome-wise analysis is usually faster than the genome-wide analysis.

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> exampleBinData

Summary: bin-level data (class: BinData)
----------------------------------------
- # of chromosomes in the data: 1
- total effective tag counts: 1637819
  (sum of ChIP tag counts of all bins)
- control sample is incorporated
- mappability score is incorporated
- GC content score is incorporated
- uni-reads are assumed
----------------------------------------
```

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

```
R> print(exampleBinData)[51680:51690,]

      chrID    coord tagCount mappability gcContent input
51680 chr21 15353100       10        1.00      0.36     4
51681 chr21 15353150       25        1.00      0.38     3
51682 chr21 15353200       61        1.00      0.39     5
51683 chr21 15353250      105        1.00      0.39     5
51684 chr21 15353300      125        1.00      0.39     6
51685 chr21 15353350      124        1.00      0.38     6
51686 chr21 15353400      109        1.00      0.38     7
51687 chr21 15353450       72        1.00      0.36     4
51688 chr21 15353500       30        0.99      0.36     2
51689 chr21 15353550       10        0.99      0.36     1
51690 chr21 15353600        6        0.99      0.36     1
```

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

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

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

Figures 1, 2, 3, 4, 5, and 6 display examples of different types of plots. As discussed in [1], we observe that mean ChIP tag count increases as mappability score increases (Figure 2). Mean ChIP tag count depends on GC score in a non-linear fashion (Figure 3). The relationship between mean ChIP tag counts and control tag counts seems to be linear, especially for small control tag counts (Figure 4). When we condition on control tag counts (Figures 5 and 6), mean ChIP tag count versus mappability and GC content relations exhibit similar patterns to that of marginal plots given in Figures 2 and 3. 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.

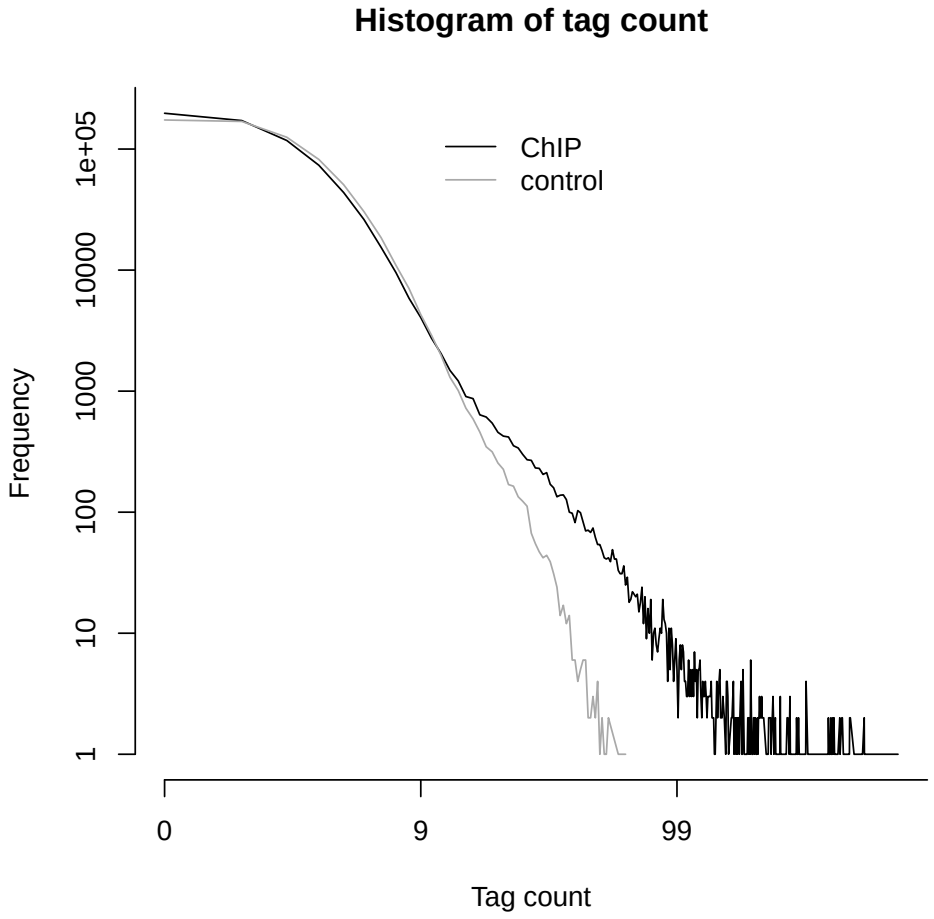

Figure 1: Histograms of the count data from ChIP and control samples.

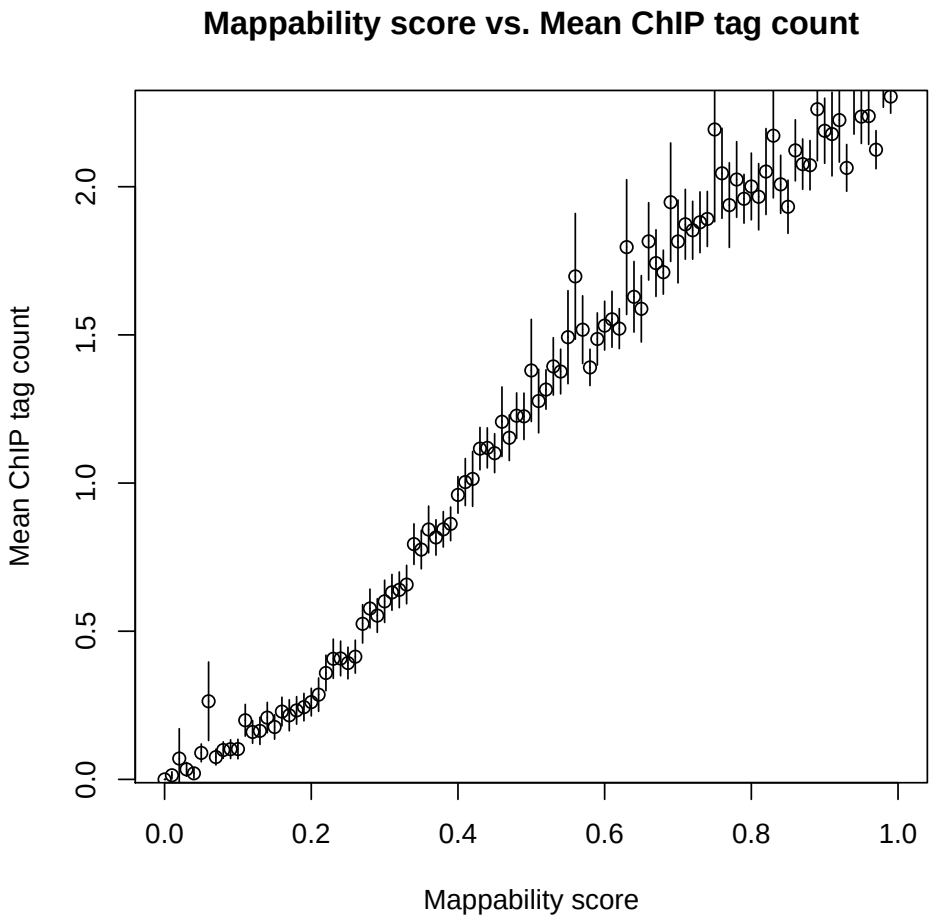

Figure 2: Mean ChIP tag count versus Mappability.

3.3 Fitting the MOSAiCS Model

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

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

'`analysisType="TS"`' 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) and it is appropriate for the data with relatively high sequencing depth. If '`bgEst=NA`' (default), '`mosaicsFit`' tries its best guess for the background estimation approach, based on the data provided. If the goodness of fit obtained using '`bgEst=NA`' is not satisfactory, we recommend to try '`bgEst="matchLow"`' and/or '`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.

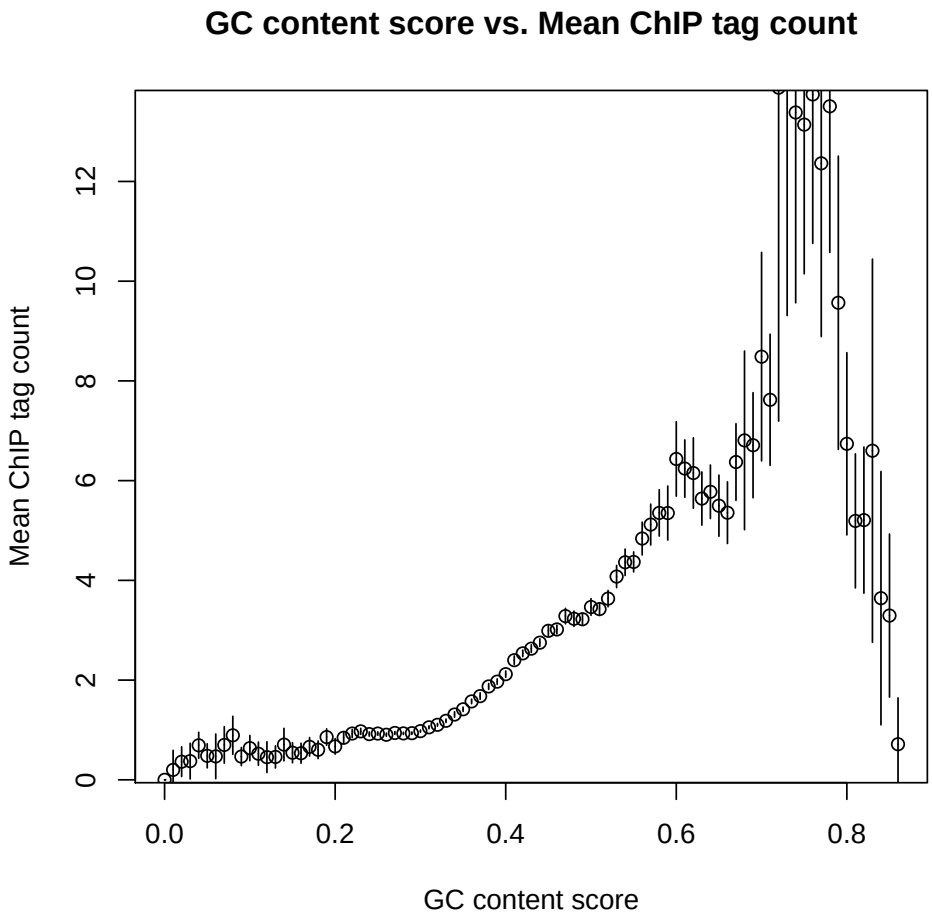

Figure 3: Mean ChIP tag count versus GC content.

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. For this dataset, we conclude that the two-signal-component model has a lower BIC and hence it provides a better fit.

```
R> exampleFit

Summary: MOSAiCS model fitting (class: MosaicsFit)
-------------------------------------------------
analysis type: two-sample analysis (with mappability & GC content)
parameters used: k = 3, meanThres = 1, s = 2, d = 0.25
BIC of one-signal-component model = 1137784
BIC of two-signal-component model = 1135762
-------------------------------------------------
```

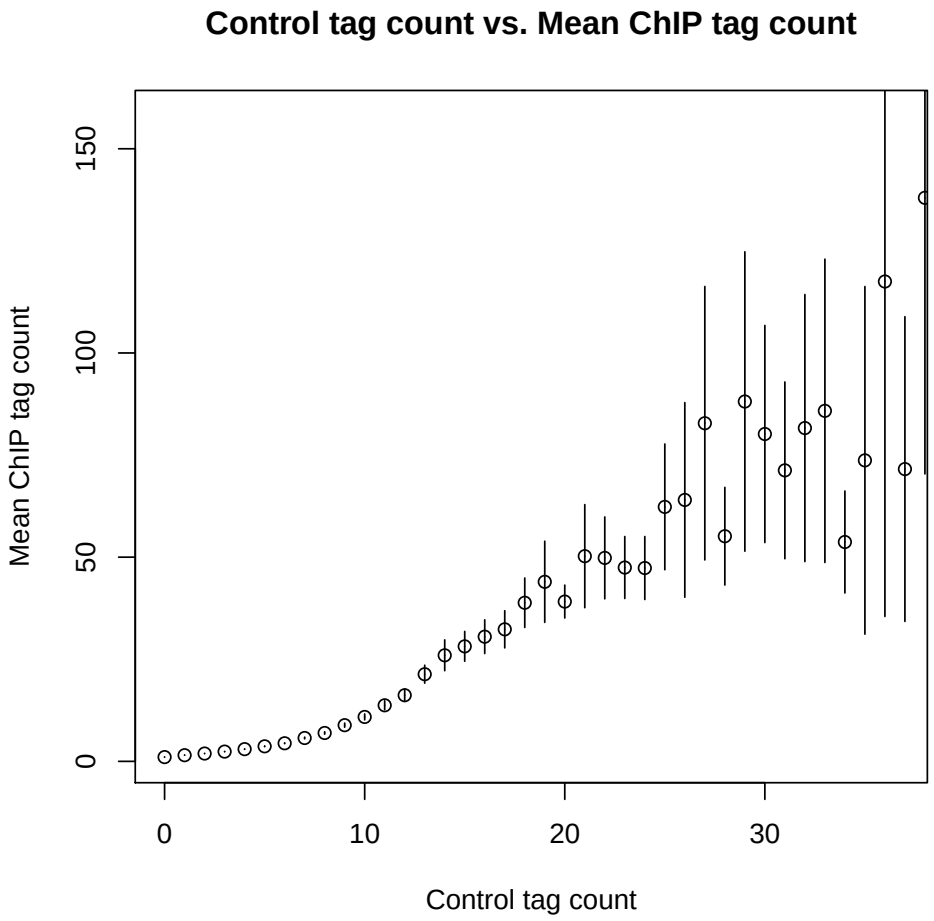

Figure 4: Mean ChIP tag count versus Control tag count.

'`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 7 displays the GOF plot for our dataset and we conclude that the two-signal-component model provides a better fit as is also supported by its lower BIC value compared to the one-signal component model.

```
R> plot(exampleFit)
```

3.4 Identifying Peaks Based on the Fitted Model

Using BIC values and GOF plots in the previous section, we concluded that two-signal-component model fits our data better. Next, we will identify peaks with the two-signal-component model at a false discovery rate (FDR) of 0.05 using the command:

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

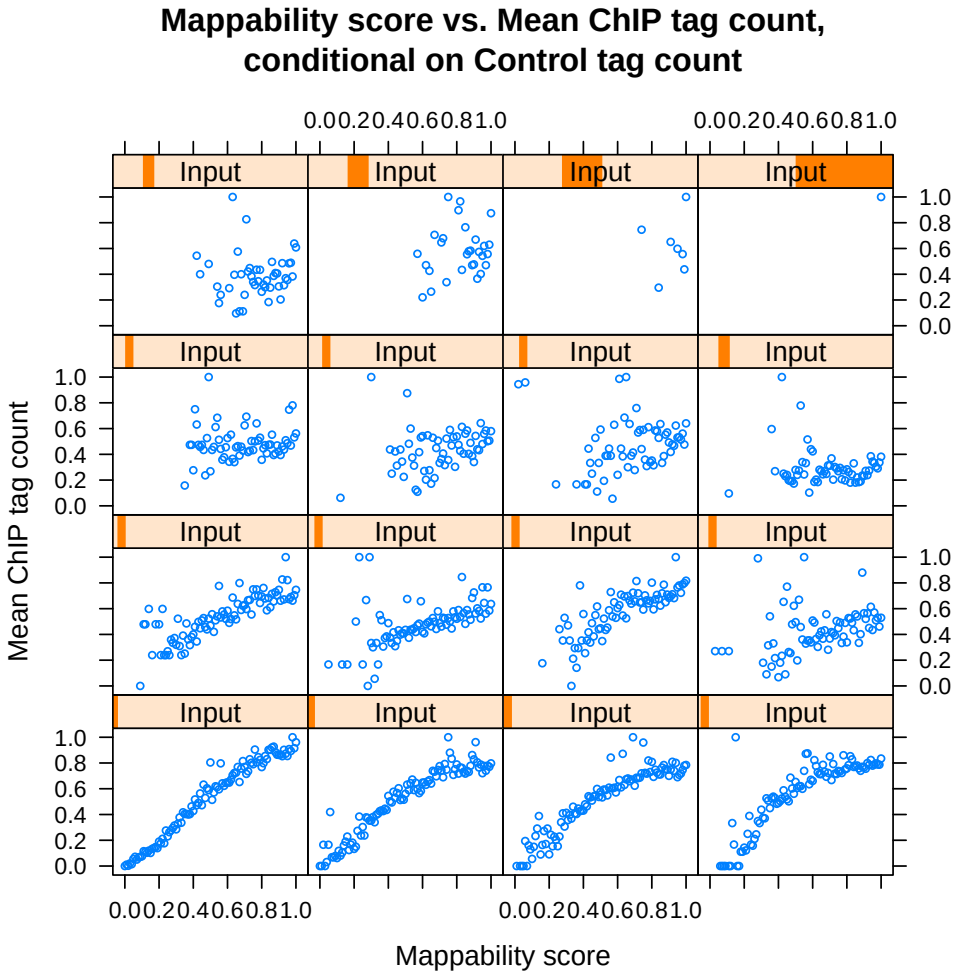

Figure 5: Mean ChIP tag count versus Mappability, conditional on control tag counts.

'`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

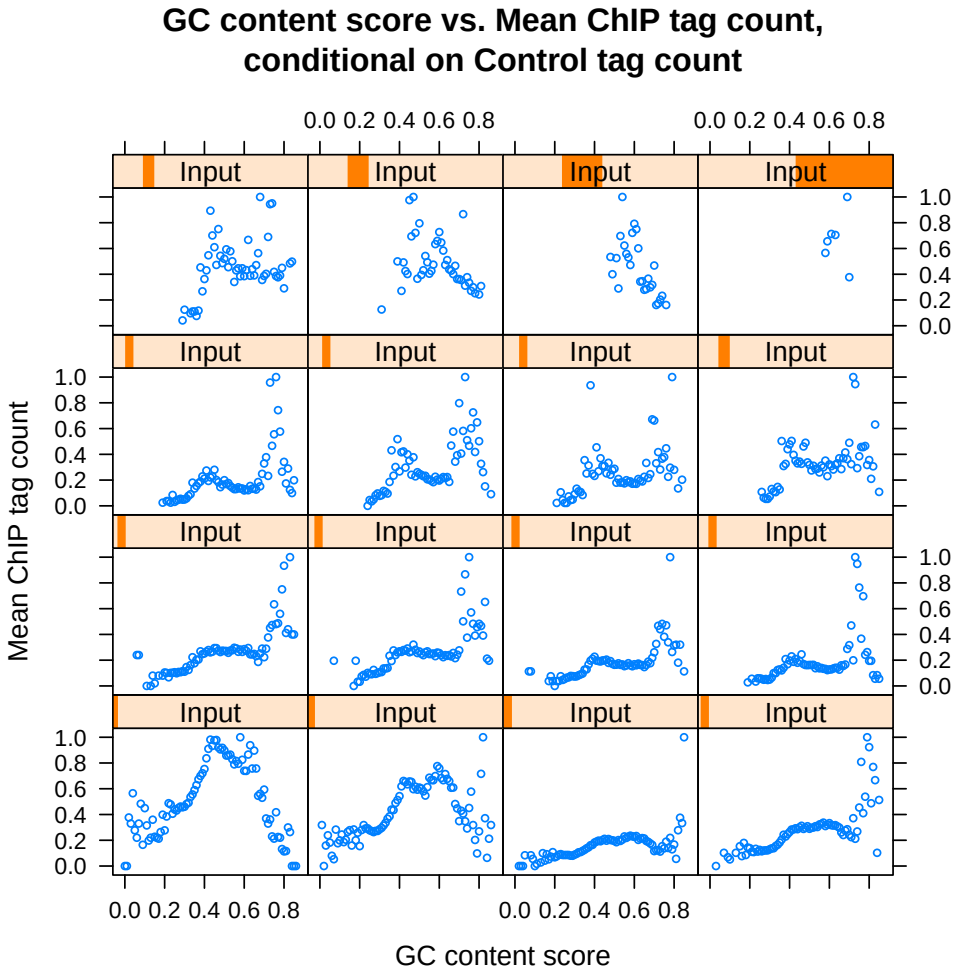

Figure 6: Mean ChIP tag count versus GC content, conditional on control tag counts.

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> examplePeak

Summary: MOSAiCS peak calling (class: MosaicsPeak)
-------------------------------------------------
final model: two-sample analysis (with M & GC) with two signal components
setting: FDR = 0.05, maxgap = 200, minsize = 50, thres = 10
# of peaks = 520
median peak width = 250
empirical FDR = 0.05
-------------------------------------------------
```

'`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

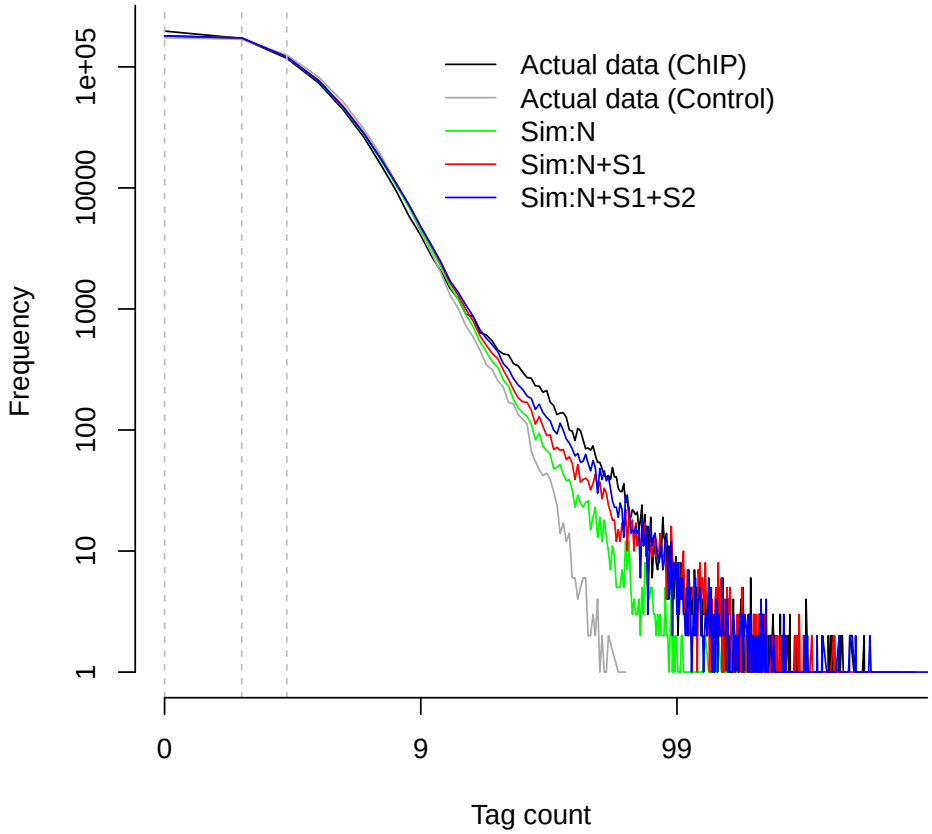

Figure 7: Goodness of Fit (GOF) plot. Depicted are actual data for ChIP and control samples with simulated data from the following fitted models: (Sim:N): Background model; (Sim:N+S1): one-signal-component model; (Sim:N+S1+S2): two-signal-component model.

different number of columns, depending on '`analysisType`' of '`mosaicsFit`'. For example, if '`analysisType="TS"`', columns are 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, averaged log base 2 ratio of ChIP over input tag counts, averaged mappability score, and averaged GC content score for each peak. Here, the posterior probability of a bin refers to the probability that the bin is not a peak conditional on data. Hence, smaller posterior probabilities provide more evidence that the bin is actually a peak.

```
R> print(examplePeak)[1:15,]
```

```
   chrID peakStart peakStop peakSize         aveP         minP aveChipCount
1  chr21  14538100 14538499      400 2.316159e-02 1.732184e-11     32.00000
2  chr21  14828000 14828449      450 4.683745e-02 4.537161e-05     21.77778
3  chr21  14901550 14901849      300 1.120661e-02 4.268164e-05     20.00000
```

```
4  chr21  15032250 15032499      250 2.122025e-02 4.995990e-04     15.00000
5  chr21  15068000 15068099      100 9.315207e-02 7.948542e-02     13.50000
6  chr21  15175200 15175299      100 8.070661e-02 3.541864e-02     15.50000
7  chr21  15177350 15177599      250 1.030375e-01 7.688995e-03     16.80000
8  chr21  15353150 15353549      400 2.190450e-06 9.188126e-26     81.37500
9  chr21  15362700 15362849      150 1.498505e-01 7.948542e-02     12.33333
10 chr21  15374650 15375349      700 6.933481e-05 2.139220e-65     88.28571
11 chr21  15378850 15379049      200 9.072711e-02 7.649026e-04     21.00000
12 chr21  15486500 15486799      300 3.124502e-02 5.137427e-04     33.00000
13 chr21  15498300 15499149      850 3.892446e-02 5.263370e-12     48.23529
14 chr21  15501950 15502249      300 2.826662e-01 7.688995e-03     14.33333
15 chr21  15502950 15503399      450 2.188426e-03 2.830815e-49     90.66667
   maxChipCount aveInputCount aveInputCountScaled aveLog2Ratio       map
1            48      2.375000            2.662827     3.140374 0.9925000
2            31      3.777778            4.235608     2.183571 1.0000000
3            25      2.833333            3.176706     2.358010 1.0000000
4            17      2.000000            2.242380     2.466431 0.9940000
5            14      2.000000            2.242380     2.160069 1.0000000
6            17      3.000000            3.363571     1.962167 0.9850000
7            20      1.800000            2.018142     2.601233 0.9800000
8           125      4.750000            5.325654     3.560025 0.9987500
9            13      1.666667            1.868650     2.240231 1.0000000
10          180      2.714286            3.043231     4.127678 1.0000000
11           23      3.750000            4.204463     2.178285 0.9975000
12           40      6.666667            7.474602     2.008631 0.9833333
13           64      7.705882            8.639760     2.346157 1.0000000
14           18      3.000000            3.363571     1.829847 1.0000000
15          144      5.111111            5.730528     3.907724 1.0000000
         GC
1  0.4250000
2  0.3788889
3  0.3933333
4  0.3320000
5  0.3900000
6  0.3800000
7  0.4540000
8  0.3787500
9  0.3800000
10 0.3557143
11 0.4325000
12 0.4016667
13 0.4311765
14 0.4333333
15 0.4644444
```

You can export peak calling results to text files in diverse file formats. Currently, '`mosaics`' package supports TXT, BED, and GFF file formats. In the exported file, TXT file format ('`type="txt"`') includes all the columns that '`print`' method returns. '`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. Peak calling results can be exported in TXT, BED, and GFF file formats, respectively, by the commands:

```
R> export( examplePeak, type="txt", fileLoc=".", fileName="TSpeakList.txt" )
R> export( examplePeak, type="bed", fileLoc=".", fileName="TSpeakList.bed" )
R> export( examplePeak, type="gff", fileLoc=".", fileName="TSpeakList.gff" )
```

'`fileLoc`' and '`fileName`' indicate the directory and the name of the exported file.

4 Two-Sample Analysis without Mappability and GC Content

Application of MOSAiCS to multiple case studies 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]. However, `mosaics` package accommodates a two-sample analysis without mappability and GC content by specification of '`analysisType="IO"`' when calling the '`mosaicsFit`' method.

```
R> inputOnlyFit <- mosaicsFit( exampleBinData, analysisType="IO" )
```

You can import bin-level data (for ChIP and control sample only) and fit MOSAiCS model for the two-sample analysis without mappability and GC content with the commands:

```
R> inputOnlyBinData <- readBins( type=c("chip","input"),
+     fileName=c( system.file( file.path("extdata","chip_chr21.txt"), package="mosaicsExample"),
+     system.file( file.path("extdata","input_chr21.txt"), package="mosaicsExample") ) )

R> inputOnlyFit <- mosaicsFit( inputOnlyBinData, analysisType="IO" )
```

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

```
R>      mosaicsRunAll(
+          chipDir="/scratch/eland/",
+          chipFileName="STAT1_ChIP_eland_results.txt",
+          chipFileFormat="eland_result",
+          controlDir="/scratch/eland/",
+          controlFileName="STAT1_control_eland_results.txt",
+          controlFileFormat="eland_result",
+          binfileDir="/scratch/bin/",
+          peakDir="/scratch/peak/",
+          peakFileName="STAT1_peak_list.txt",
+          peakFileFormat="txt",
+          reportSummary=TRUE,
+          summaryDir="/scratch/reports/",
+          summaryFileName="mosaics_summary.txt",
+          reportExploratory=TRUE,
+          exploratoryDir="/scratch/reports/",
+          exploratoryFileName="mosaics_exploratory.pdf",
```

```
+          reportGOF=TRUE,
+          gofDir="/scratch/reports/",
+          gofFileName="mosaics_GOF.pdf",
+          byChr=FALSE,
+          FDR=0.05,
+          fragLen=200,
+          binSize=fragLen,
+          capping=0,
+          bgEst=NA,
+          signalModel="BIC",
+          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', 'fragLen', 'binSize', and 'capping'. See Section 3.3' for details of the argument 'bgEst'. See Section 3.4' for details of the arguments 'FDR', 'signalModel', 'peakFileFormat', 'maxgap', 'minsize', and 'thres'.

5 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:

```
R> OneSampleBinData <- 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") ) )

------------------------------------------------------------
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
------------------------------------------------------------
```

the file name of a control dataset in 'fileName' here. In order to fit a MOSAiCS model for the one-sample analysis, you need to specify 'analysisType="OS"' when calling the 'mosaicsFit' method.

Table 1: **Summary of the arguments of 'mosaicsRunAll' method.** In the tables (a) and (b), each cell in the second to fourth columns of the second to fourth rows corresponds to each argument. In table (c), the first column corresponds to arguments.

(a) Input and output files

Category	Directory	File name	File format
ChIP sample	chipDir	chipFileName	chipFileFormat
Matched control sample	controlDir	controlFileName	controlFileFormat
Bin-level files	binfileDir		
Peak list	peakDir	peakFileName	peakFileFormat

(b) Reports

Category	Generate report?	Directory	File name
Analysis summary	reportSummary *	summaryDir	summaryFileName
Exploratory plots	reportExploratory *	exploratoryDir	exploratoryFileName
GOF plots	reportGOF *	gofDir	gofFileName

* 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)?
fragLen	Average fragment length.
binSize	Bin size.
capping	Cap read counts in aligned read files?
bgEst	Background estimation approach.
signalModel	Signal model.
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.
nCore	Number of CPUs used for parallel processing/computing. **

** Relevant only when multicore package is installed.

```
R> OneSampleFit <- mosaicsFit( OneSampleBinData, analysisType="OS" )
```

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

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

6 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/`).

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] 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.