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.

May 13, 2011

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

We assume that you already have the aligned files for your samples (ChIP and control) such as files obtained from the ELAND aligner. R package '`mosaics`' takes bin-level data as input and these bin-level data can easily be generated from the aligned data using the preprocessing scripts we provide. You can download these preprocessing scripts from the '`mosaics`' package companion website, `http://www.stat.wisc.edu/~keles/Software/mosaics/`. You also need to have 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 Reading Bin-Level Data into the R Environment

'mosaics' package assumes chromosome-wise analysis of ChIP-seq data. For the two-sample analysis, 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'.

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 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 coordinates = 683452
- 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, ]
```

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

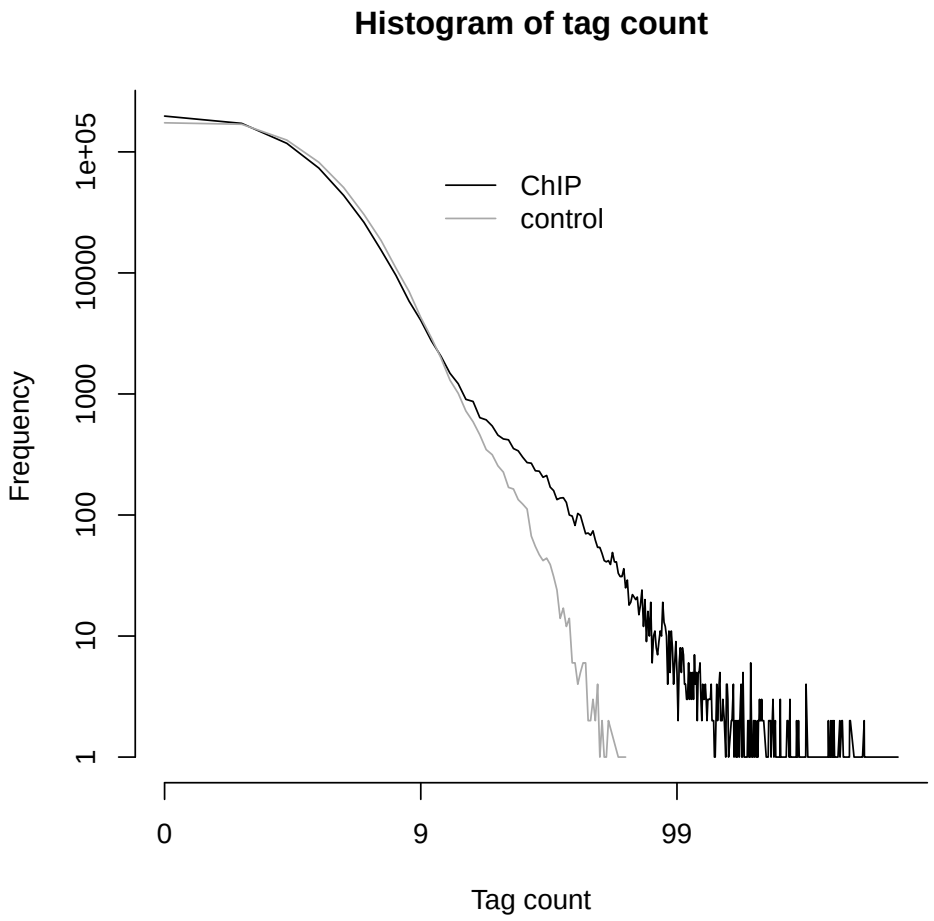

Figure 1: Histograms of the count data from ChIP and control samples.

3.2 Fitting MOSAiCS

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")
```

'`analysisType="TS"`' indicates implementation of the two-sample analysis. '`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. For this dataset, we conclude that the two-signal-component model has a lower BIC and hence it provides a better fit.

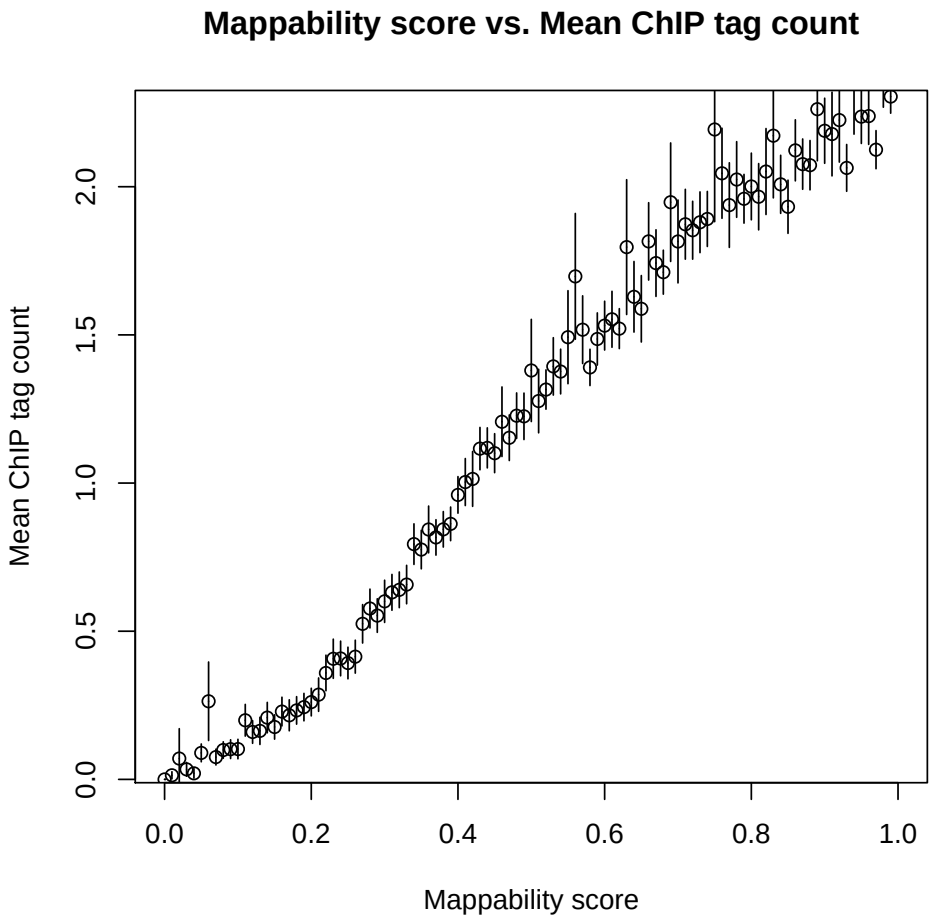

Figure 2: Mean ChIP tag count versus Mappability.

```
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 = 1185504
BIC of two-signal-component model = 1181909
-------------------------------------------------
```

'`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)
```

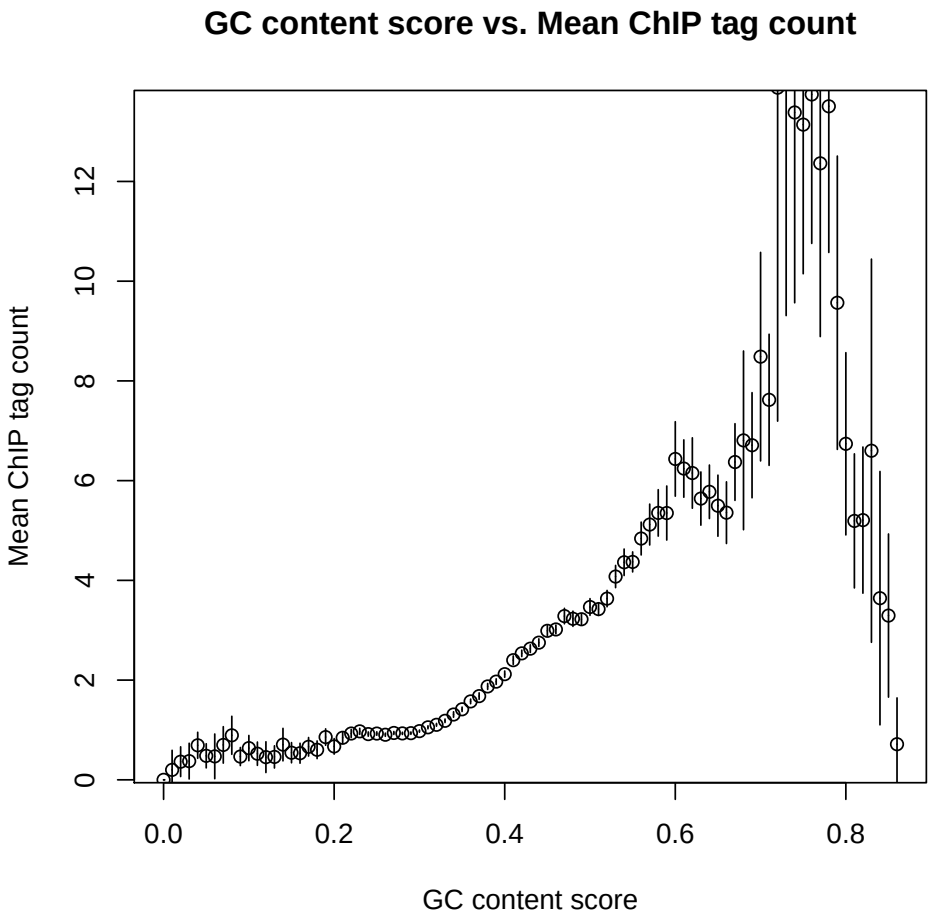

Figure 3: Mean ChIP tag count versus GC content.

In addition to '`analysisType`', '`mosaicsFit`' method provides parameters to tune the background distribution of the MOSAiCS model. We specified appropriate default values for these parameters based on computational experiments and analysis of diverse ChIP-seq datasets. Default values work well in general but some tuning might be required for some cases. You may need to consider parameter tuning if the fitted background model is too similar to the actual data in the GOF plot or you encounter some warning or error messages while running '`mosaicsFit`' method. Section 6 provides basic guidelines on parameter tuning. 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`.

3.3 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:

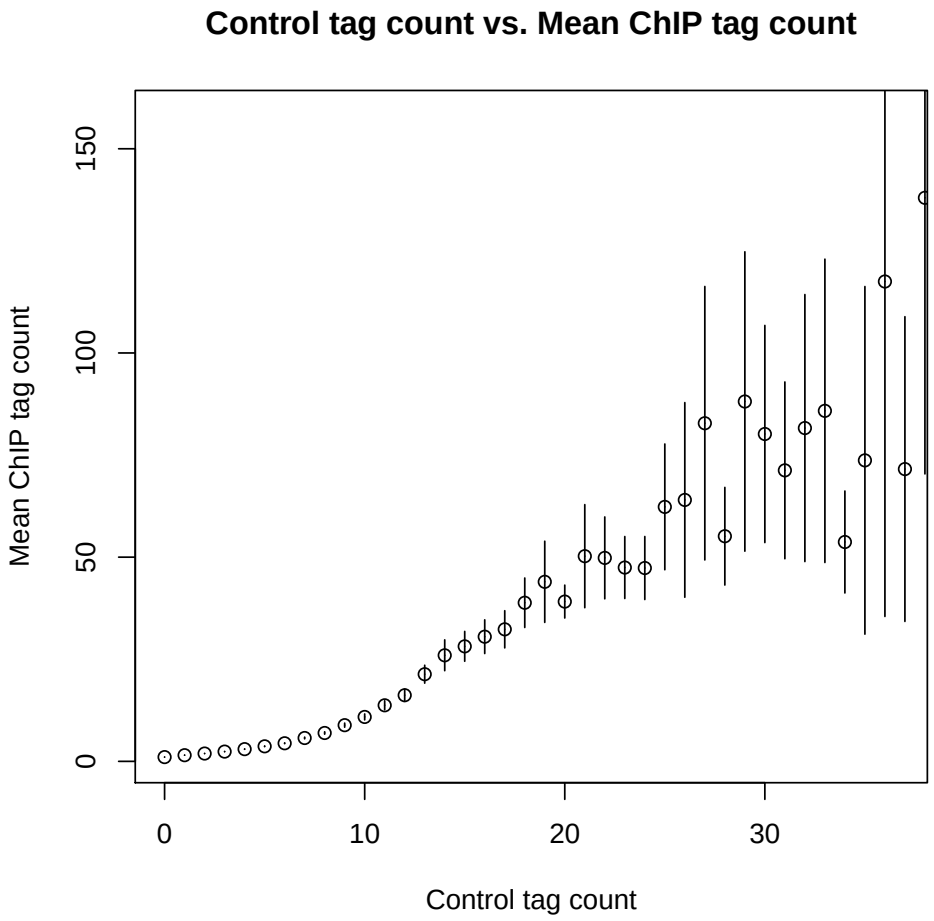

Figure 4: Mean ChIP tag count versus Control tag count.

```
R> examplePeak <- mosaicsPeak(exampleFit, 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`'. 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

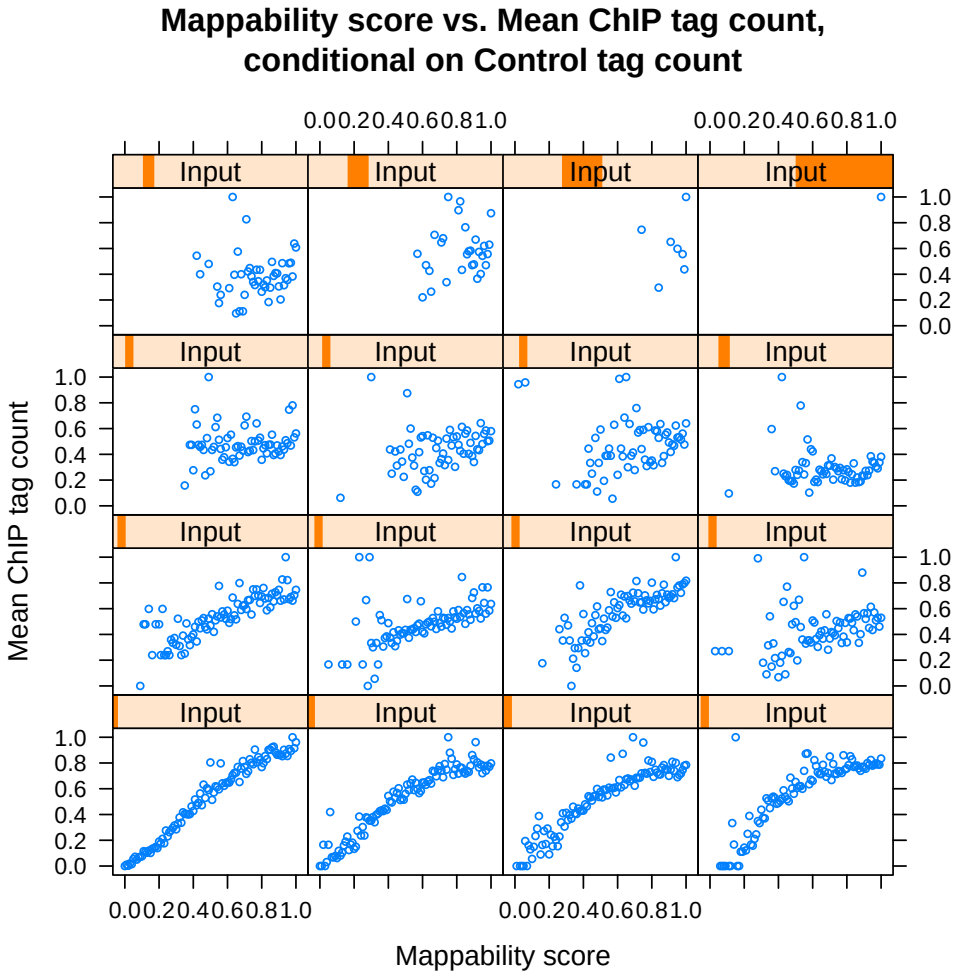

Figure 5: Mean ChIP tag count versus Mappability, conditional on control tag counts.

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> 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 = 988
median peak width = 249
empirical FDR = 0.0545
-------------------------------------------------
```

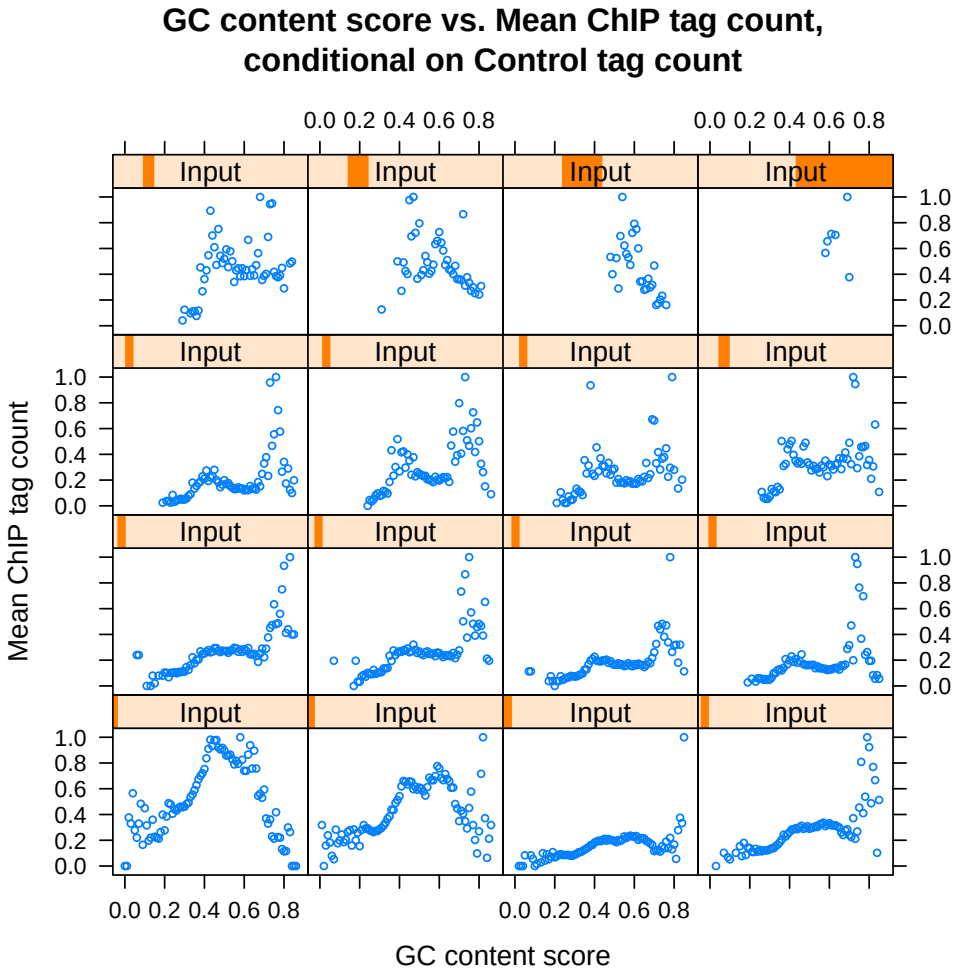

Figure 6: Mean ChIP tag count versus GC content, conditional on control tag counts.

'`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, 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, ]
```

```
   peakStart peakStop peakSize         aveP         minP aveChipCount
1    9732700  9732899      199 3.836228e-01 1.100792e-01     10.50000
2    9811600  9811699       99 7.993523e-02 4.979124e-02     11.50000
3    9815400  9815699      299 1.722436e-01 1.115138e-02     12.50000
```

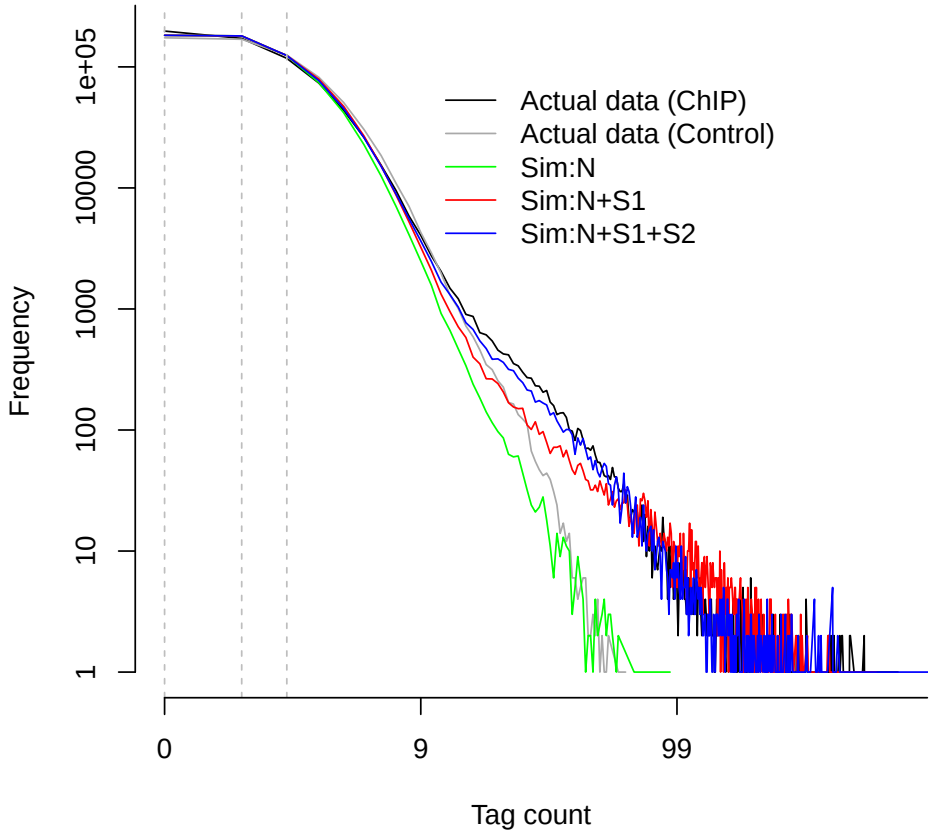

Figure 7: Goodness of Fit (GOF) plot. Depicted are actual data for ChIP and control samples with simulated data from the following fitted models: (Sim:N): Background model; (Sim:N+S1): one-signal-component model; (Sim:N+S1+S2): two-signal-component model.

```
4    9817800  9817899       99 1.085175e-01 2.079744e-02     12.50000
5   10144350 10144449       99 1.051044e-01 8.131156e-02     11.50000
6   14538100 14538549      449 1.772463e-02 5.545260e-16     29.77778
7   14828000 14828499      499 1.342458e-02 1.770893e-06     20.70000
8   14901550 14901849      299 6.075843e-04 2.613464e-07     20.00000
9   15032250 15032499      249 7.595103e-04 4.043507e-05     15.00000
10  15068000 15068199      199 8.088829e-02 9.975310e-03     12.25000
11  15172050 15172199      149 3.969889e-02 6.904682e-03     11.66667
12  15175200 15175399      199 5.437793e-03 3.214256e-03     17.25000
13  15177350 15177699      349 3.215324e-02 1.550662e-04     16.14286
14  15353150 15353549      399 1.091832e-08 2.939206e-39     81.37500
15  15358300 15358649      349 7.929136e-02 1.488240e-04     19.00000
   maxChipCount aveInputCount aveInputCountScaled aveLog2Ratio       map
1            11      4.000000            4.484761     1.105108 0.7350000
```

```
2            12      3.000000            3.363571     1.517192 0.9650000
3            16      3.666667            4.111031     1.393560 0.9250000
4            13      3.500000            3.924166     1.463428 0.8750000
5            12      1.000000            1.121190     2.557827 0.8250000
6            48      2.222222            2.491534     3.082062 0.9933333
7            31      3.700000            4.148404     2.111159 0.9990000
8            25      2.833333            3.176706     2.358010 1.0000000
9            17      2.000000            2.242380     2.466431 0.9940000
10           14      2.000000            2.242380     2.023989 1.0000000
11           13      1.333333            1.494920     2.370160 1.0000000
12           19      4.000000            4.484761     1.780164 0.9325000
13           20      2.142857            2.402551     2.394367 0.9857143
14          125      4.750000            5.325654     3.560025 0.9987500
15           21      6.142857            6.887311     1.361290 1.0000000
          GC
1  0.3400000
2  0.4400000
3  0.4850000
4  0.4400000
5  0.4400000
6  0.4244444
7  0.3780000
8  0.3933333
9  0.3320000
10 0.3950000
11 0.3800000
12 0.3925000
13 0.4685714
14 0.3787500
15 0.6900000
```

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",
+     chrID = "chr21")
R> export(examplePeak, type = "bed", fileLoc = ".", fileName = "TSpeakList.bed",
+     chrID = "chr21")
R> export(examplePeak, type = "gff", fileLoc = ".", fileName = "TSpeakList.gff",
+     chrID = "chr21")
```

'`fileLoc`' and '`fileName`' indicate the directory and the name of the exported file. '`chrID`' means chromosome ID and you need to specify this because '`mosaics`' package assumes chromosome-wise analysis and does not require user to supply chromosome ID during the analysis. The word specified in '`chrID`' will appear in the first column in the exported file.

4 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
------------------------------------------------------------
```

Note that you don't need to provide '`"input"`' in '`type`' and 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"`' instead of '`analysisType="TS"`' when calling the '`mosaicsFit`' method.

```
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)
```

5 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("
+     "chip_chr21.txt"), package = "mosaicsExample"), system.file(file.path("extdata",
+     "input_chr21.txt"), package = "mosaicsExample")))

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

6 Tuning of MOSAiCS Parameters

In the two-sample analysis, users can control three tuning parameters: '`s`', '`d`', and '`meanThres`'. '`s`' and '`d`' are parameters of the background distribution and control the functional form used for the control data. Please see [1] for further details on these two model parameters. '`meanThres`' controls the number of strata used at the robust linear regression modelling step of the background distribution fitting. '`mosaics`' package uses the following parameter setting as default:

```
R> exampleFit <- mosaicsFit(exampleBinData, analysisType = "TS",
+     meanThres = 1, s = 2, d = 0.25)
```

Users might need to consider parameter tuning especially when the fitted background model is too similar to the actual data, resulting in too few peaks. If such cases are detected or predicted, '`mosaicsFit`' prints out warning or error messages. You may also be able to detect this case using the GOF plot. Using a higher '`s`' value and lower '`meanThres`' often solves the problem, e.g., '`s = 6`' and '`meanThres = 0`'.

7 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, JA Thomson, R Stewart, and S Keleş (2010), "A Statistical Framework for the Analysis of ChIP-Seq Data", submitted (`http://works.bepress.com/sunduz_keles/19/`).

[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.