The gcapc user's guide

Mingxiang Teng mxteng@jimmy.harvard.edu
Rafael A. Irizarry rafa@jimmy.harvard.edu
Department of Biostatistics, Dana-Farber Cancer Institute,
Harvard T.H. Chan School Public Health, Boston, MA, USA

2016-10-19

Contents

1 Introduction

ChIP-seq has been widely utilized as the standard technology to detect protein binding regions, where peak calling algorithms were developed particularly to serve the analysis. However, existing peak callers lack of power on ranking peaks' significance due to sequencing technology might undergo sequence context biases, *e.g.* GC bias. gcapc is designed to address this deficiency by modeling GC effects into peak calling.

The gcapc R-package performs GC bias estimation, peak calling, and plots on intermediate results. It requires the inputs as one BAM file for ChIP-seq as well as other optional parameters. A common analysis contains four steps.

1. Reads coverage. In this step, BAM file records will be converted to coverages on basepair resolution for forward and reverse strands separately.
2. Binding width estimation. This parameter is a measurement for the size of protein binding region in crosslinked complexes of ChIP experiments.
3. GC effects estimation. Generalized linear mixture models followed by EM algorithms are performed to evaluate potential GC effects.
4. Peak calling. Enrichment scores are evaluated by permutation analysis for significance. Peaks are reported with enrichment scores and p-values.

2 Getting Started

Load the package in R

```
library(gcapc)
```

3 Preparing Inputs

The inputs could be as minimum as a path to a BAM file, which is an indexed alignment records for sequencing reads. However, additional options are encouraged to be specified to accelerate the analysis and improve the accuracy. The following set are the options which can be customized by users.

1. BAM records filtering options. In the function *rc5end*, reads can be filtered for selected chromosomes, mapping quality, duplicate removal, etc. Downstream analysis could be highly accelerated if only a subset of chromosomes are analyzed. This actually provides a divide and conquer strategy if one ChIP-seq experiment is extremely deeply sequenced.
2. Sequencing fragments options. If one has prior knowledge on the size of sequencing fragments. The optional arguments in function *bdwidth* could be specified to limit searching in narrower ranges; Or, this function can be omitted if binding width are known in advance. Note that this binding width might not be equivalent to the binding width of protein in biology, since it could be affected by crosslinking operations.
3. Sampling size for GC effects estimation. The default is 0.05, which means 5% of genome will be used if analysis is based on whole genome. However, for smaller genomes or small subset of chromosomes, this size should be tuned higher to ensure accuracy. In the other hand, larger size results longer GC effects estimation.
4. EM algorithm priors and convergence. Options for EM algorithms can be tuned to accelerate the iterations.
5. Permutation times. As we suggested in the function help page, a proper times of permutation could save time as well as ensuring accuracy.

In this vignette, we will use enbedded file *chipseq.bam* as one example to illustrate this package. This file contains about ~80000 reads from human chromosome 21 for CTCF ChIP-seq data.

```
bam <- system.file("extdata/chipseq.bam",package="gcapc")
```

4 Peak Calling

For details of peak calling algorithms, please refer to our paper (Teng and Irizarry 2016).

4.1 Reads coverage

The first step is to generate the reads coverage for both forward and reverse strands. The coverage is based on single nucleotide resolution and uses only the 5' end of BAM records. That means, if duplicates are not allowed, the maximum coverage for every nucleotide is 1.

```
cov <- rc5end(bam)
cov
## $fwd
## RleList of length 1
## $chr21
## integer-Rle of length 48129895 with 40225 runs
##   Lengths: 9414767       1    8350       1 ...     116       1   41437
##   Values :       0       1       0       1 ...       0       1       0
##
##
## $rev
## RleList of length 1
```

```
## $chr21
## integer-Rle of length 48129895 with 40427 runs
##   Lengths: 9412972       1    3087       1 ...     367       1   34767
##   Values :       0       1       0       1 ...       0       1       0
```

Obejct *cov* is a two-element list representing coverages for forward and reverse strands, respectively, while each element is a list for coverages on individual chromosomes.

4.2 Binding width

The second step is to estimate the binding width of ChIP-seq experiment. This step could be omitted if binding width is known in advance. Binding width is further treated as the size of region unit for weighted GC bias estimation and peak calling.

```
bdw <- bdwidth(cov)
## Starting to estimate bdwidth.
## ...... cycle 1 for bind width estimation
## ...... cycle 2 for bind width estimation
## ...... cycle 3 for bind width estimation
## ...... estimated bind width as 110
bdw
## [1] 110
```

If additional information is known from sequencing fragments, this step could be speeded up. For example, narrowing down the range size helps.

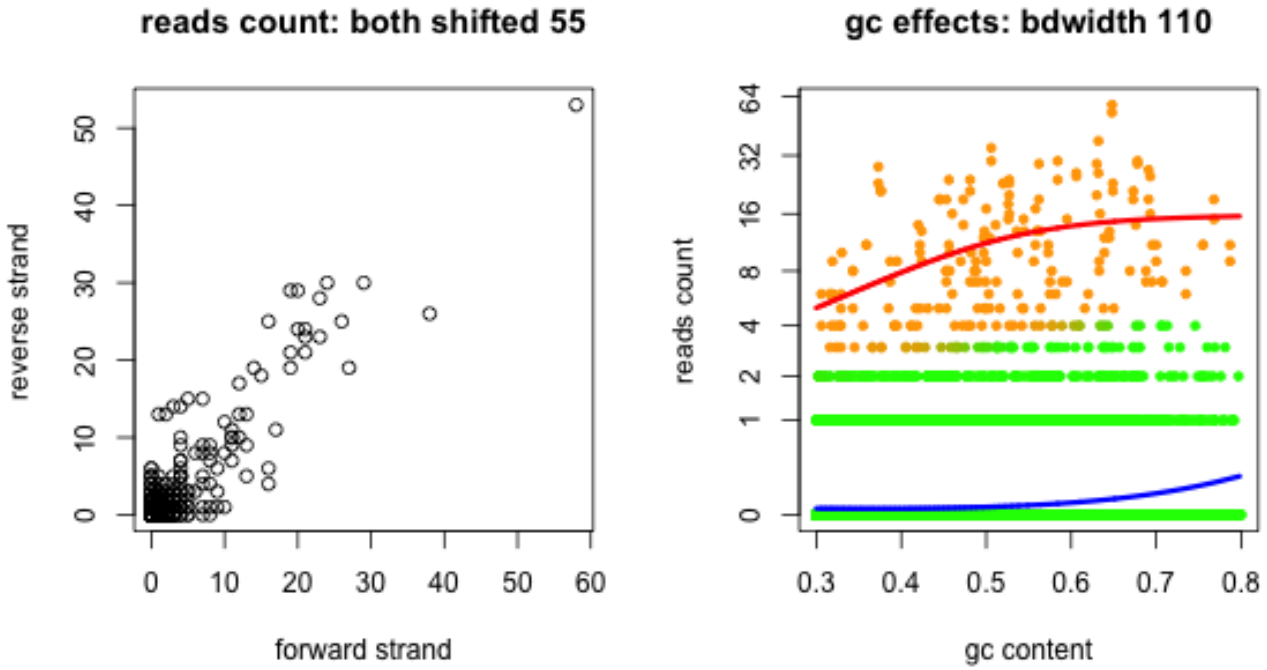
```
bdw <- bdwidth(cov,range=c(50L,150L),step=10L)
## Starting to estimate bdwidth.
## ...... cycle 1 for bind width estimation
## ...... cycle 2 for bind width estimation
## ...... estimated bind width as 110
bdw
## [1] 110
```

4.3 GC effects

This step performs GC effects estimation using the proposed models. It is noted that by allowing to display the plots, one can view intermediate results which provide you direct sense on your ChIP-seq data, such as the extent of GC effects. Also, the EM algorithms iterations are enabled by default to display the trace of log likelihood changes, and other notification messages are printed for courtesy.

```
gcb <- gcbias(cov,bdw,samp=0.15,plot=TRUE)
## Starting to estimate GC effects.
## ...... sampling regions
## ......... estimating using 65631 regions
## ...... counting reads
## ...... calculating GC content with flanking 55
## ...... estimating GC effects
## ......... iteration 1    ll -18288.28    increment 68121.55
## ......... iteration 2    ll -17755.65    increment 532.6311
## ......... iteration 3    ll -17728.45    increment 27.19517
## ......... iteration 4    ll -17728.94    increment -0.4933942
## ......... iteration 5    ll -17730.08    increment -1.131432
```

```
## ......... iteration 6    ll -17730.7     increment -0.6250008
## ......... iteration 7    ll -17730.77    increment -0.07090929
```



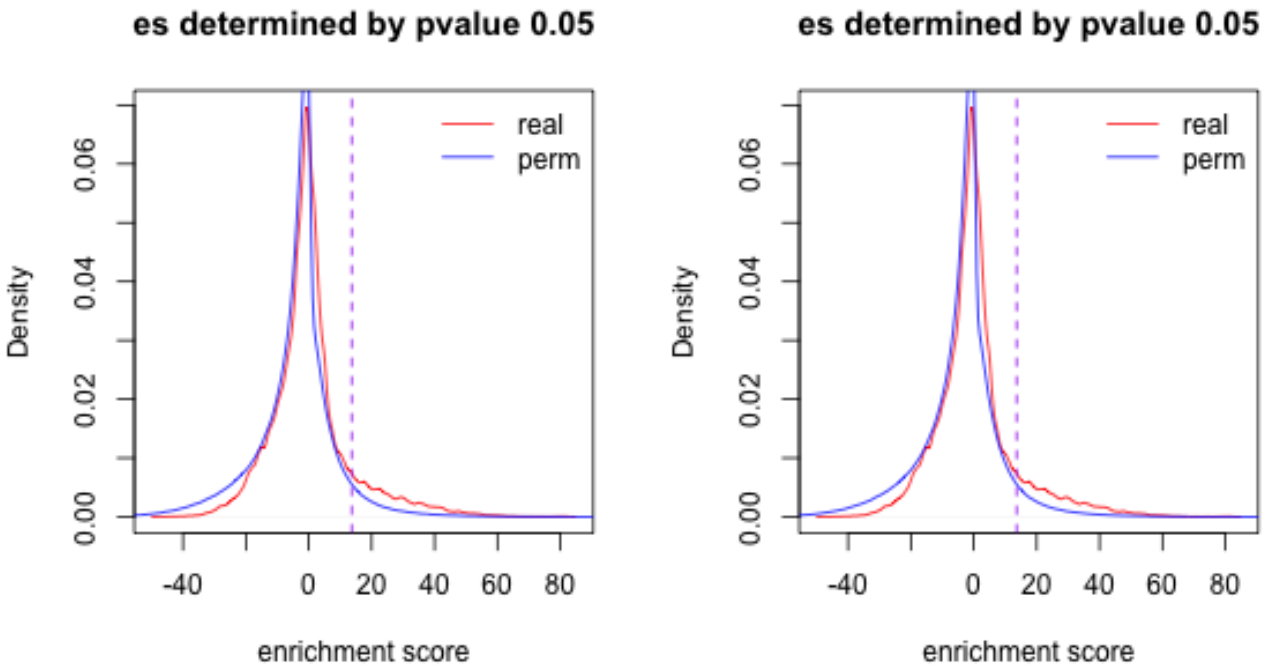
Here, the left figure provides the correlation between forward and reverse strands signals, by using the estimated binding width as region unit. The right figure shows the raw and predicted GC effects using miture model. The effect for the background regions will be utilized in downstream analysis.

4.4 Peak significance

This is the last step of gcapc. It uses information generated in previous steps, calculates enrichment scores and performs permutation analysis to propose significant peak regions. Final peaks are formated into *GRanges* object, and meta columns are used to record significance. Additional notification messages are also printed.

```
layout(matrix(1:2,1,2))
peaks <- callpeaks(cov,gcb,bdw,plot=TRUE)
## Starting to call peaks.
## ...... prefiltering regions
## ...... caculating GC content
## .
## ...... caculating GC effect weights
## ...... estimating enrichment score
## ...... permutation analysis
## ...... reporting peaks
## ......... enrichment scores cut at 13.68802
## ......... ploting enrichment scores
## ......... reporting peak bumps
## ......... summarizing peak score and pvalue
peaks <- callpeaks(cov,gcb,bdw,plot=TRUE,permute=20L)
## Starting to call peaks.
## ...... prefiltering regions
```

```
## ...... caculating GC content
## .
## ...... caculating GC effect weights
## ...... estimating enrichment score
## ...... permutation analysis
## ...... reporting peaks
## ......... enrichment scores cut at 13.65605
## ......... ploting enrichment scores
## ......... reporting peak bumps
## ......... summarizing peak score and pvalue
peaks
## GRanges object with 248 ranges and 2 metadata columns:
##         seqnames              ranges strand |               es
##            <Rle>           <IRanges>  <Rle> |        <numeric>
##     [1]    chr21 [ 9827281,  9827468]      * | 18.3375142966309
##     [2]    chr21 [15626039, 15626186]      * | 28.7080280216165
##     [3]    chr21 [15632135, 15632288]      * |  17.638853117982
##     [4]    chr21 [16110372, 16110527]      * | 32.7756401344712
##     [5]    chr21 [16146228, 16146383]      * | 39.7183704197612
##     ...      ...                 ...    ... .              ...
##   [244]    chr21 [47393613, 47393772]      * | 40.6537903131998
##   [245]    chr21 [47563496, 47563681]      * | 37.9585894055569
##   [246]    chr21 [47567395, 47567541]      * | 42.2541691000438
##   [247]    chr21 [47573257, 47573407]      * | 38.8278568188772
##   [248]    chr21 [48081182, 48081308]      * | 34.1353386838699
##                          pv
##                   <numeric>
##     [1]  0.0307930255763877
##     [2]  0.0115385221364425
##     [3]  0.0330176986502809
##     [4] 0.00803634262906361
##     [5] 0.00428680216375188
##     ...                 ...
##   [244] 0.00395304868441781
##   [245] 0.00504359014757627
##   [246] 0.00341526180347673
##   [247] 0.00465836878315218
##   [248] 0.00710860774119004
##   -------
##   seqinfo: 1 sequence from an unspecified genome; no seqlengths
```



It is noted that two tests using different number of permutation times results almost the same cutoff on enrichment scores, which suggests small number of permutations are allowed to save time. The left figure shows here the cutoff on enrichment scores based on 10 times of permutations, and right figure shows it based on 20 times of permutations.

5 Summary

In this vignette, we went through main functions in this package, and illustrated how they work. By easily following these steps, users could call peaks based on ChIP-seq data. Note that this package is not limited to protein binding ChIP-seq experiments. It can be used in Histone studies as well, since the protein binding width in this algorithm is actually a feature of crosslinked complex instead of real biological protein binding.

References

Teng, Mingxiang, and Rafael A. Irizarry. 2016. "Accounting for GC-Content Bias Reduces Systematic Errors and Batch Effects in ChIP-Seq Peak Callers." *Submitted*.