The chromstaR user's guide

Aaron Taudt*

April 1, 2016

Contents

1 Introduction

ChIP-seq has become the standard technique for assessing the genome-wide chromatin state of DNA. *chromstaR* provides functions for the joint analysis of multiple ChIP-seq samples. It allows peak calling for transcription factor binding and histone modifications with a narrow (e.g. H3K4me3, H3K27ac, ...) or broad (e.g. H3K36me3, H3K27me3, ...) profile. All analysis can be performed on each sample individually (=univariate), or in a joint analysis considering all samples simultaneously (=multivariate).

2 Outline of workflow

Every analysis with the *chromstaR* package starts from aligned reads in either BAM or BED format. In the first step, the genome is partitioned into non-overlapping, equally sized bins and the reads that fall into each bin

*aaron.taudt@gmail.com

are counted. These read counts serve as the basis for both the univariate and the multivariate peak- and broad-region calling. Univariate peak calling is done by fitting a three-state Hidden Markov Model to the binned read counts. Multivariate peak calling for $\mathcal{S}$ samples is done by fitting a $2^{\mathcal{S}}$-state Hidden Markov Model to all binned read counts.

2.1 Task 3: Finding combinatorial chromatin states

Most experimental studies that probe several histone modifications are interested in combinatorial chromatin states. An example of a simple combinatorial state would be [H3K4me3+H3K27me3], which is also frequently called "bivalent promoter", due to the simultaneous occurrence of the promoter marking H3K4me3 and the repressive H3K27me3. Finding combinatorial states with *chromstaR* is equivalent to a multivariate peak calling. The following code chunks demonstrate how to find bivalent promoters and do some simple analysis:

```
library(chromstaR)
```

```
## === Step 1: Binning ===
# Let's get some example data for H3K27me3 and H3K4me3 in rat liver
file.path <- system.file("extdata","euratrans", package='chromstaRData')
bedfiles <- list.files(file.path, pattern="liver.*H3K4me3|liver.*H3K27me3",
                       full.names=TRUE)
# This is only necessary for BED files. BAM files are handled automatically.
data(rn4_chrominfo)
# We use bin size 1000bp and chromosome 12 to keep the example quick
binned.data <- list()
for (bedfile in bedfiles) {
  binned.data[[basename(bedfile)]] <- binReads(bedfile, format='bed', binsize=1000,
                                          assembly=rn4_chrominfo, chromosomes='chr12')
}
```

```
## === Step 2: Univariate peak calling ===
# The univariate fit is obtained for each replicate
models <- list()
for (i1 in 1:length(binned.data)) {
  models[[i1]] <- callPeaksUnivariate(binned.data[[i1]], ID=names(binned.data)[i1],
                                      max.time=60)
}
```

```
## === Step 3: Constructing the combinatorial states ===
# This step is only necessary if you have replicates for each sample.
# To ensure that replicates are treated as such, and not as independent
# samples, we have to construct the proper combinatorial states:
```

```r
# First, we get all the marks (we could specify them by hand, but we are lazy)
IDs <- names(binned.data)
marks <- sapply(strsplit(IDs,'-'),'[[',2)
print(marks)
```

```
## [1] "H3K27me3" "H3K27me3" "H3K27me3" "H3K4me3"  "H3K4me3"  "H3K4me3"
```

```r
# Second, we obtain the combinatorial states
# Look up ?stateBrewer on how to use this function
states <- stateBrewer(marks)
print(states)
```

```
##        combination state
## 1                     0
## 2          H3K4me3     7
## 3         H3K27me3    56
## 4 H3K27me3+H3K4me3    63
```

```r
## === Step 4: Multivariate peak calling ===
multi.model <- callPeaksMultivariate(models, use.states=states, eps=1, max.time=60)
```

```
## HMM: number of states = 4
## HMM: number of bins = 46782
## HMM: maximum number of iterations = none
## HMM: maximum running time = 60 sec
## HMM: epsilon = 1
## HMM: number of experiments = 6
##  Iteration              log(P)             dlog(P)    Time in sec
##          0                -inf                  -              0
## HMM: Precomputing densities ...
##  Iteration              log(P)             dlog(P)    Time in sec
##          0                -inf                  -              0
##          1      -629970.204822                inf              0
##          2      -628146.574559         1823.630263              0
##          3      -627956.663999          189.910560              0
##          4      -627909.013321           47.650678              0
##          5      -627894.914770           14.098550              0
##          6      -627889.957698            4.957072              0
##          7      -627887.910231            2.047467              0
##          8      -627886.945380            0.964851              0
## HMM: Convergence reached!
## HMM: Recoding posteriors ...
```

```r
## === Step 5: Export to genome browser ===
# Export combinatorial states
exportMultivariate(multi.model, filename='your-file', what=c('peaks','counts'))
exportMultivariate(multi.model, filename='your-combstates', what=c('combstates'))
```

```r
## === Step 6: Enrichment analysis ===
# Get coordinates of rat genes
```

```
library(biomaRt)
ensembl <- useMart('ENSEMBL_MART_ENSEMBL', host='may2012.archive.ensembl.org',
                   dataset='rnorvegicus_gene_ensembl')
genes <- getBM(attributes=c('ensembl_gene_id', 'chromosome_name', 'start_position',
                            'end_position', 'strand', 'external_gene_id'),
               mart=ensembl)
# Transform to GRanges for easier handling
genes <- GRanges(seqnames=paste0('chr',genes$chrom),
                 ranges=IRanges(start=genes$start, end=genes$end),
                 strand=genes$strand,
                 ID=genes$ensembl_gene_id, name=genes$external_gene_id)
print(genes)

## GRanges object with 29516 ranges and 2 metadata columns:
##           seqnames                ranges strand |                 ID
##              <Rle>             <IRanges>  <Rle> |        <character>
##       [1]    chr13     [1120899, 1121213]      - | ENSRNOG00000043314
##       [2]    chr13     [1192186, 2293551]      - | ENSRNOG00000031539
##       [3]    chr13     [3174383, 3175216]      + | ENSRNOG00000028603
##       [4]    chr13     [4377731, 4379174]      - | ENSRNOG00000030028
##       [5]    chr13     [4866302, 4866586]      - | ENSRNOG00000040235
##       ...      ...                    ...    ... .                ...
##   [29512]     chr6 [134310258, 134310338]      + | ENSRNOG00000035137
##   [29513]     chr9 [  6920889,   6921049]      - | ENSRNOG00000035407
##   [29514]    chr11 [ 40073746,  40073816]      - | ENSRNOG00000043706
##   [29515]     chr2 [233090372, 233090478]      - | ENSRNOG00000035272
##   [29516]     chr6 [ 92917449,  92917541]      + | ENSRNOG00000045521
##               name
##           <character>
##       [1]   LOC682397
##       [2]   LOC304725
##       [3]
##       [4]  D3ZPH4_RAT
##       [5]  F1LZC7_RAT
##       ...         ...
##   [29512]    SNORD113
##   [29513]          U1
##   [29514]    SNORD19B
##   [29515]          U6
##   [29516]
##   -------
##   seqinfo: 22 sequences from an unspecified genome; no seqlengths
```

3 FAQ

4 Session Info

```
sessionInfo()

## R Under development (unstable) (2016-02-17 r70182)
```

```
## Platform: x86_64-pc-linux-gnu (64-bit)
## Running under: Ubuntu 14.04.4 LTS
##
## locale:
##  [1] LC_CTYPE=en_US.UTF-8       LC_NUMERIC=C
##  [3] LC_TIME=nl_NL.UTF-8        LC_COLLATE=en_US.UTF-8
##  [5] LC_MONETARY=nl_NL.UTF-8    LC_MESSAGES=en_US.UTF-8
##  [7] LC_PAPER=nl_NL.UTF-8       LC_NAME=C
##  [9] LC_ADDRESS=C               LC_TELEPHONE=C
## [11] LC_MEASUREMENT=nl_NL.UTF-8 LC_IDENTIFICATION=C
##
## attached base packages:
## [1] stats4    parallel  stats     graphics  grDevices utils     datasets
## [8] methods   base
##
## other attached packages:
##  [1] biomaRt_2.27.2        chromstaR_0.98.0      chromstaRData_0.99.0
##  [4] ggplot2_2.1.0         GenomicRanges_1.23.25 GenomeInfoDb_1.7.6
##  [7] IRanges_2.5.40        S4Vectors_0.9.44      BiocGenerics_0.17.3
## [10] knitr_1.12.3          devtools_1.10.0
##
## loaded via a namespace (and not attached):
##  [1] Rcpp_0.12.4                 AnnotationDbi_1.33.7
##  [3] XVector_0.11.7              magrittr_1.5
##  [5] GenomicAlignments_1.7.20    zlibbioc_1.17.1
##  [7] BiocParallel_1.5.21         munsell_0.4.3
##  [9] colorspace_1.2-6            highr_0.5.1
## [11] stringr_1.0.0               plyr_1.8.3
## [13] tools_3.3.0                 SummarizedExperiment_1.1.22
## [15] grid_3.3.0                  Biobase_2.31.3
## [17] gtable_0.2.0                DBI_0.3.1
## [19] digest_0.6.9                reshape2_1.4.1
## [21] formatR_1.3                 bitops_1.0-6
## [23] RCurl_1.95-4.8              RSQLite_1.0.0
## [25] memoise_1.0.0               evaluate_0.8.3
## [27] stringi_1.0-1               Rsamtools_1.23.6
## [29] Biostrings_2.39.12          scales_0.4.0
## [31] XML_3.98-1.4

warnings()

## NULL
```