Carmen M. Livi

ChIP-seq quality Control

ChIC

A short introduction

Contents

1 Introduction

ChIP-seq quality **C**ontrol package (ChIC) provides functions and data structures to assess the quality of ChIP-seq samples [1].

In particular, we introduce a new set of quantitative quality control (QC) metrics, the Local enrichment profile Metrics (LM), by defining quantitative scores that describe the shape of ChIP-seq enrichment profiles. These metrics are not replacing, but instead extending, previously proposed quantitative metrics for scoring ChIP-seq data, that are also computed as part of a comprehensive set. These include the recommended ENCODE QC metrics (EM) and metrics describing the global enrichment (GM) in the ChIP-seq experiment. This comprehensive set of metrics is leveraged to build a machine learning classifier to obtain a single score reliably summarizing the data quality and asssessing the quality of ChIP-seq data. This random forest based quality control score is named `ChIC RF-score`.

This package is meant to be used in conjunction with the `ChIC.data` package that contains a reference compendium with QC metrics pre-computed on thousands of ChIP-seq samples, as well as the pre-computed machine learning classifiers, that can also be used as a reference for easier evaluation of new datasets.

The `ChIC` package provides a user friendly wrapper function (`chicWrapper()`) to compute all of the the QC metrics, the `ChIC RF-score`, summary plots for QC-metrics, plots for the comparison of metagene profiles against reference profiles, and plots for the the comparison of single QC-metrics against the compendium values. In addition to the user-friendly wrapper, the pacakge contains several additional functionalities for each analysis step by step for more experienced users.

We recommend referring to the manuscript preprint by Livi *et al.* [1] for a more thorough discussion of the method rationale and for its benchmarking against previous solutions for ChIP-seq quality control.

1.1 Overview of ChIC analysis worflow

The ChIC package analysis workflow includes multiple steps that can be summarized as follows:

1. The input data (BAM files) are read into R objects.
2. A comprehensive set of quantitative QC-metrics are computed. These include:
   (a) ENCODE Metrics (EM): a set of QC metrics derived from ENCODE recommended best practices [2].
   (b) Global enrichment profile Metrics (GM): a set of quantitative metrics quantitative scores that describe the shape of ChIP-seq enrichment profiles derived from the global distribution of enrichment profiles. These are mostly quantitative metrics derived fromt the the so-called "fingerprint plot" [3, 4].

(c) Local enrichment profile metrics (LM) metrics: a set of quantitative scores that describe the shape of ChIP-seq enrichment profiles. We use the so-called "metagene" profiles, i.e. the average ChIP-seq signal over a set of genes, to derive a large set of quantitative features [1].

3. A set of summary plots is drawn to show how the ChIP-seq sample analyzed compares to the reference compendium of ChIP-seq datasets in terms of:

   (a) The distribution of selected individual quality metrics

   (b) The shape of enrichment profile around the annotated gene body, the annotated transcription start sites (TSS) or annotated transcript end (TES).

4. ChIC RF score: a single score summary of the sample quality with respect to the reference compendium. This score is based on a random forest machine learning classifier (see below and [1]).

The analysis steps, as well as the package functions to perform them, are described more in details in the following sections.

2 ChIC analysis in one command with `chicWrapper()`

2.1 Example input data

The ChIC package takes as input aligned high throughput sequencing read files in the BAM file format (.bam). For any ChIP-seq sample under investigation, the end user is expected to provide a pair of bam files: one for the chromatin immunoprecipitation experiment (ChIP) and one for the control experiment (input). For the neophytes, it must be noted that the ChIP-seq filed adopted a potentially misleading terminoligy as the control experiments consituted by crosslinked and sonicated chromatin (without immunoprecipitation) are generally called "input control" experiments, but most often they are just referred to as "input" experiment. This may cause some confusion with respect to the "input" data to any given software or algorithm. In the context of this vignette we will try to alwasy specify "input control" whenever the terminology may be misleading. We may just refer to ChIP/input pair as in this case the pairing of the two types of samples make it clear the seond one is the "input control".

In this Vignette tutorial we will illustrate ChIC functionalities by using a ChIP-seq sample from the ENCODE project data portal (ID: ENCFF000BFX) and its matched input control (ID: ENCFF000BDQ). This specific ChIP-seq sample immunoprecipitation target was histone H3 lysine 4 trimethylation (H3K4me3), which is a histone post translational modification (histone mark) generally associated to the promoter of transcribed genes. The BAM files containing the alignment to human genome (hg19 build) can can be downloaded from:

https://www.encodeproject.org/files/ENCFF000BFX/
https://www.encodeproject.org/files/ENCFF000BDQ/

In the example R code chunk below we are using a call to the system tool "wget" to download the data in the current working directory

```
> library(ChIC)
> chipName <- "ENCFF000BFX"
> inputName <- "ENCFF000BDQ"
> system(paste("wget https://www.encodeproject.org/files/",
+     chipName, "/@@download/",
+     chipName, ".bam", sep=""))
> system(paste("wget https://www.encodeproject.org/files/",
+     inputName, "/@@download/",
+     inputName, ".bam", sep=""))
```

2.2 The `chicWrapper()` function

The user friendly `"chicWrapper()"` function is a single command allowing the end users to run ChIC analysis on a ChIP/control pair of BAM files. The wrapper will take care of reading the BAM files, computing all the QC metrics, run the machine learning model to compute the `ChIC RF-score`.

```
> ChIC_RFscore<-chicWrapper(
+     chipName=chipName,
+     inputName=inputName,
+     savePlotPath=getwd(),
+     target= "H3K4me3",
+     read_length=36,
+     annotationID="hg19"
+     )
```

The function returns as ouput on the command line the random forest based ChIC RF score. The score ranges from 0 to 1 (0 for the worst prediction and 1 for perfect one). Assessing data quality can be described as a binary classification problem with the QC-metrics as independent input variables and the data quality as the dependent output variable (class labels: good or poor quality). We built random forest based classifiers to discriminate ChIP-seq samples with good vs poor enrichment. In order to train the random forest models, for each ChIP sample in our reference compendium we simulated a "failed" counterpart with lower lower signal-to-noise by specifically down-sampling aligned reads located within enrichment peaks.

Distinct random forest models were trained for different types of immunoprecipitation targets, as distincr chromatin mark types are expected to yield enrichment peaks with different shapes: see [1] for more details on the classifiers design. As such it is important to specify which individual protein or

chromatin mark was targeted for ChIP-seq immunoprecipitation in the experiment, and/or to consider where a more generic class of chromatin mark should be used as reference for the random forest classifier. This choice is defined with the `"target"` parameter. Pre-computed random forest models foo various targets and chromatin mark categories are stored in the `ChIC.data` package. See also the paragraph below on `ChIP targets categories` for more details on the available pre-computed models.

Other parameters that should be mentioned are the `"read_length"` parameter, that is generally known to the users based on the sequencing machine run settings, but can also be inferred from an inspection of the BAM file or of the original raw sequencing data (FASTQ file). This parameter is used to set some analysis thresholds in the "phantom peak" analysis of cross-correlation profiles [2, 5–7].

The `"annotationID"` parameter refers to the reference genome build used for aligning the sequencing reads provided in the input BAM files. Currently supported options are "hg19", "hg38", "mm9", "mm10" and "dm3". Please note that these reference annotations are used to process the input BAM files to compute the genome/species specific metagene profiles used to compute the QC metrics. The pre-computed random forest models were trained on a compendium of human ChIP-seq samples. These can in principle be used also to assess the quality of ChIP-seq in other organisms as long as the selected targes is expected to have a similar metagene profile across species: see also cross-species test cases in [1].

Finally, the `"savePlotPath"` parameter defines the path to save into a single PDF a summary report containing all the QC plots related to `ChIC` analysis. In the current version of the `"chicWrapper()"` function this parameter must be set to a value different than NULL. If it is null, the file will be saved in the current working directory as for the default option. If the `"savePlotPath"` parameter contains a path to a directroy, the saved filename will be built as `"chipName"_"inputName"_ChIC_report.pdf`. If the `"savePlotPath"` parameter contains a full path to a filename, the user specified filename will be used instead for the output summary PDF.

2.3 ChIP target categories

As described in the accompanying manuscript [1], we trained distinct machine learning classifiers for specific classes of ChIP targets. These resulted in distinct random forest models for:

- Chromatin marks with "sharp" enrichment peaks (Sharp category)
- Chromatin marks with "broad" enrichment peaks (Broad category)
- RNA Polymerase 2, which has a dinstinctive mixed sharp/broad enrichment profile (Pol2 category)
- Transcription factors and/or other proteins with specific DNA binding domains and very localized binding peaks (TF category)

As discussed in the manuscript [1], the "Broad" chromatin marks categody is generally the more challenging one for assessing the quality of ChIP-seq data. This is due to the fact that this category actually contains marks with very different distributions and enrichment profile shapes. As such, for three specific "Broad" chroamtin marks that are frequently profiled in literature we derived specific random forest models for each of them, including:

- H3K36me3
- H3K27me3
- H3K9me3

When specifying the `"target"` parameter in `chicWrapper()`, the end users will need to keep in mind this categorization of ChIP targets as that will have an effect on

1. The selection of the random forest model used to compute the ChIC RF score.
2. The selection of the reference compendium subset of samples used to generaty summary plot for QC metrics and metagene profiles.

2.3.1 ChIP target categories - model selection in `chicWrapper()`

When the `"target"` parameter in `chicWrapper()` is specified, the function will adopt the more "specific" random forest model, if available. This means that, when specifying H3K36me3, or H3K27me3, or H3K9me3, the histone mark specific random forest model will be used to compute the ChIC RF score.

For all of the other histone marks, the random forest model of the corresponding general category will be used. For example, if the target H3K4me3 is specified, then the random forest model for "Sharp" chromatin marks will be used. We can verify which chromatin marks of the reference compendium were grouped within each category by using the function `listAvailableElements()`.

For example to list all "Sharp" chromatin marks:

```
> listAvailableElements("sharp")
```

```
 [1] "H3K27ac"   "H3K9ac"    "H3K14ac"   "H2BK5ac"
 [5] "H4K91ac"   "H3K18ac"   "H3K23ac"   "H2AK9ac"
 [9] "H3K4me3"   "H3K4me2"   "H3K79me1"  "H2AFZ"
[13] "H2A.Z"     "H4K12ac"   "H4K8ac"    "H3K4ac"
[17] "H2BK12ac"  "H4K5ac"    "H2BK20ac"  "H2BK120ac"
[21] "H2AK5ac"   "H2BK15ac"
```

and to list all "Broad" chroamtin marks

```
> listAvailableElements("broad")
```

```
 [1] "H3K23me2" "H3K9me2"  "H3K9me3"  "H3K27me3" "H4K20me1"
 [6] "H3K36me3" "H3K56ac"  "H3K9me1"  "H3K79me2" "H3K4me1"
[11] "H3T11ph"
```

It should be noted that we sub-grouped chromatin mark samples by the expected shape of their enrichment peaks into “Sharp” and “Broad” following the ENCODE3 guidelines (https://www.encodeproject.org/chip-seq/histone/).

If the end user wishes to apply the general random forest model for "broad" or "sharp" chromatin marks with another ChIP target not included in this list, this can be achieved by just specifying the `target="broad"` or `target="sharp"` parameter in the call to `chicWrapper()`, respectively.

Likewise, the list of trasncription factors that has been considered in the reference compendium is accessible with `listAvailableElements("TF")`. If the end users is targeting a transcription factor, or other protein with specific DNA binding domains and very localized binding peaks which is not included in this list, it is anyway possible to apply the random forest model for transcription factors. This is achieved by using the `target="TF"` parameter in `chicWrapper()`.

2.3.2 ChIP target - reference compendium subset selection for summary plots in `chicWrapper()`

The definition of the `"target"` parameter in `chicWrapper()` will also affect the selection of the datasets used to draw summary plots of the reference metaprofiles.

The end users will be able to plot the metaprofiles for the ChIP-seq (query) sample under examination with `chicWrapper()`. However, the additional metaprofile plots where the query ChIP-seq sample is comapred to the average profile observed in the reference compendium datasets will be available only if the specified target is among the available ones that can be displayed with the `listAvailableElements()` function as described above.

Thus, for the comaprison of metaprofiles to the "avereage" the end users will not be allowed to specify a "generic" category (borad/sharp/TF) which instead are available for the random forest model. As such, if one the general categories is indicated, the coresponding random forest model will be sued to compute the ChIC RF score, the metaprofiles for the "query" ChIP-seq sample will be reporte din the sumamry, but the comaprison to the average metaprofile of the specific mark will not be included in the output PDF.

2.4 Summary PDF report

The `chicWrapper()` will return the ChIC RF score as output in the R command line, and also write on the specified output directory a PDF with a set of summary plots. These will include (in this order):

1. The cross-correlation profile along with the main metrics derived from this plot (e.g. the RSC score) [1, 2].

2. The fingerprint plot
3. The metagene profile (average profile over all genes) around the TSS, with separate lines for reads density of ChIP and input control samples
4. The metagene profile around the TSS for normalized log2(ChIP/input) enrichment
5. The metagene profile around the TES for reads density of Chip and input control samples
6. The metagene profile around the TES for normalized log2(ChIP/input) enrichment
7. The metagene profile over the gene body and flanking regions (rescaled to plot together geens of different sizes) with reads density of Chip and input control samples
8. he metagene profile over the gene body for normalized log2(ChIP/input) enrichment

If the `"target"` parameter is set to one of the available ones within the reference compendium (the full list can be displayed with the `listAvailableElements()` function as described above), then additional plots comparing the average profiles against the ones in the compendium are reported as well in teh PDF. These additional pages in the output summary PDF will include:

9. The RSC score for the query sample, compared to the frequency distribution of RSC values in samples of the same class of chromatin mark in the reference compendium
10. The TSS metagene profile for ChIP reads density compared to mean profile (+/-2stdErr) for the same chromatin mark samples in the reference compendium
11. The TSS metagene profile for input reads density compared to mean profile (+/-2stdErr) of the reference compendium
12. The TSS metagene profile for normalized log2(ChIP/input) enrichment compared to mean profile (+/-2stdErr) of the reference compendium
13. The TES metagene profile for ChIP reads density compared to mean profile (+/-2stdErr) of the reference compendium
14. The TES metagene profile for input reads density compared to mean profile (+/-2stdErr) of the reference compendium
15. The TES metagene profile for normalized log2(ChIP/input) enrichment compared to mean profile (+/-2stdErr) of the reference compendium
16. The gene body metagene profile for ChIP reads density compared to mean profile (+/-2stdErr) of the reference compendium

17. The gene body metagene profile for input reads density compared to mean profile (+/-2stdErr) of the reference compendium

18. The gene body metagene profile for normalized log2(ChIP/input) enrichment compared to mean profile (+/-2stdErr) of the reference compendium

3 ChIC analysis step by step

In the section above we have seen how to run the ChIC analysis with a single user friendly command. In this section we are going to examine more in details the individual analysis steps which are also included in the `chicWrapper()` function. However, each of these steps allows additional analyses and reporting options that more experienced R users can easily adopt to further enrich the output of ChIC.

3.1 Reading BAM files

In the first analysis step, the input BAM files are read into R objects storing the aligned reads positions and their quality scores into a `"taglist"` object as defined by the spp package [5].

Plase note that the `readBamFile()` function illustrated in the code chunk below will expect as input a BAM filename without the `".bam"` extension that will be automatically added by the function. The filename can also contain the pathname if the data file is not located in the current working directory.

```
> chipBam <- readBamFile(chipName)
> inputBam <- readBamFile(inputName)
```

PLEASE NOTE: In order to comply with time limits for example code chunks to be executed when compiling Vignettes for Bioconductor and R packages, from now on we will actually use a smaller subset of these data. Namely, for the subsequent analysis steps we will use only chromosomes 17, 18 and 19 from the samples described above.

```
> subset_chromosomes<-c("chr17","chr18","chr19")
> chipSubset<-lapply(chipBam,
+     FUN=function(x) {x[subset_chromosomes]})
> inputSubset<-lapply(inputBam,
+     FUN=function(x) {x[subset_chromosomes]})
```

These `"chipSubset"` and `"inputSubset"` objects are also preloaded and available in the companion `"ChIC.data"` package and can be easily imported in the workspace as follows:

```
> data("chipSubset", package = "ChIC.data",
+     envir = environment())
> str(chipSubset)
```

```
List of 2
 $ tags   :List of 3
  ..$ chr17: num [1:465615] -195 344 440 470 494 527 559 -716 ...
  ..$ chr18: num [1:190568] 11518 11553 -11658 11712 ...
  ..$ chr19: num [1:291668] 61042 61064 61067 61067 ...
 $ quality:List of 3
  ..$ chr17: int [1:465615] 6 75 75 69 75 75 78 69 ...
  ..$ chr18: int [1:190568] 23 51 75 60 42 30 41 75 ...
  ..$ chr19: int [1:291668] 0 0 0 0 0 0 0 0 ...
```

```
> data("inputSubset", package = "ChIC.data",
+  envir = environment())
> str(inputSubset)
```

```
List of 2
 $ tags   :List of 3
  ..$ chr17: num [1:290602] -274 614 -976 1368 ...
  ..$ chr18: num [1:246863] -10170 -10308 10399 10455 ...
  ..$ chr19: num [1:141065] -60298 60277 60294 60427 ...
 $ quality:List of 3
  ..$ chr17: int [1:290602] 70 73 22 74 74 44 74 64 ...
  ..$ chr18: int [1:246863] 15 0 46 0 0 73 56 19 ...
  ..$ chr19: int [1:141065] 0 0 0 0 0 0 0 0 ...
```

3.2 Computing ENCODE Metrics (EM)

Then we proceed with computing the first set of quantitative quality control metrics, which is derived from the recommendation by ENCODE consortium [2]. The set of ENCODE Metrics (EM) as described in [1] can be computed in ChIC by a convenient wrapper function `"qualityScores_EM()"`.

```
> EM_Results <- qualityScores_EM(
+     chipName=chipName,
+     inputName=inputName,
+     chip.data=chipSubset,
+     input.data=inputSubset,
+     annotationID="hg19",
+     read_length=36)
```

The main parameters are again the same (with the same meaning and value) as described above for the "chicWrapper" function. In addition, the "chip.data" and "input.data" are optional parameters that can be used to pass as input a pair of R objects already containing the aligend reads inported into a "taglist" as described above.

If the "chip.data" and "input.data" are omitted (defaulte value is "NULL"), the texttt"qualityScores_EM()" function will take care of importing the aligned reads data from the BAM files specfied in "chipName" and "inputName", similarly to waht described above.

Here we are taking advantage of the posibility to provide as input a pair of "taglist" R objects, so that we can perform these analyses on the smaller subset of data (only chromosomes 17, 18 and 19), so as to allow a faster execution fo the vignette examples.

Please note that the "qualityScores_EM()" is actually performing many analyses in background, including performing peaks calling with multiple alternative parameters and algorithms as implemented in the spp package [5]. The end users can speed up the ChIC analysis by taking adavanted of parallel computations on mulitple computing cores in the CPU of their machine. This can be achieved by simply specifying a number of cores to be used in the analysis with the "mc" parameter (default value is 1). The "mc" parameter is available in all of the "wrapper" functions implemented in ChIC. If a number larger than 1 is specified, the software will try to use the user defined number of CPUs/cores for the analysis steps allowing parallelization.

```
> EM_Results <- qualityScores_EM(
+     chipName=chipName,
+     inputName=inputName,
+     chip.data=chipSubset,
+     input.data=inputSubset,
+     annotationID="hg19",
+     read_length=36,
+     mc=20)
```

3.2.1 EM cross correlation profile and QC metrics

The "qualityScores_EM()" function produces the Cross-correlation plot (see Figure 1) and returns a number of QC-metrics [1]. The "savePlotPath" parameters defines the path in which the Cross-Correlation plot (as pdf) should be saved. If nothing is provided the plot will be forwarded to default DISPLAY. For a more detailed discussion about the meaning of the cross correlation profile and its associated scores we must refer the user to the reference literature [2, 6, 7].

The output value, stored in "EM_Results" in the example code above, is actually a list object containing multiple EM scores. Amongst others the tag.shift value is an analysis parameter that will be passed as inptu to subsequent analyses step as well. This is the half of the average fragment size as inferred from the cross-correlation profile (see also [5]), i.e. the value that is used also in some

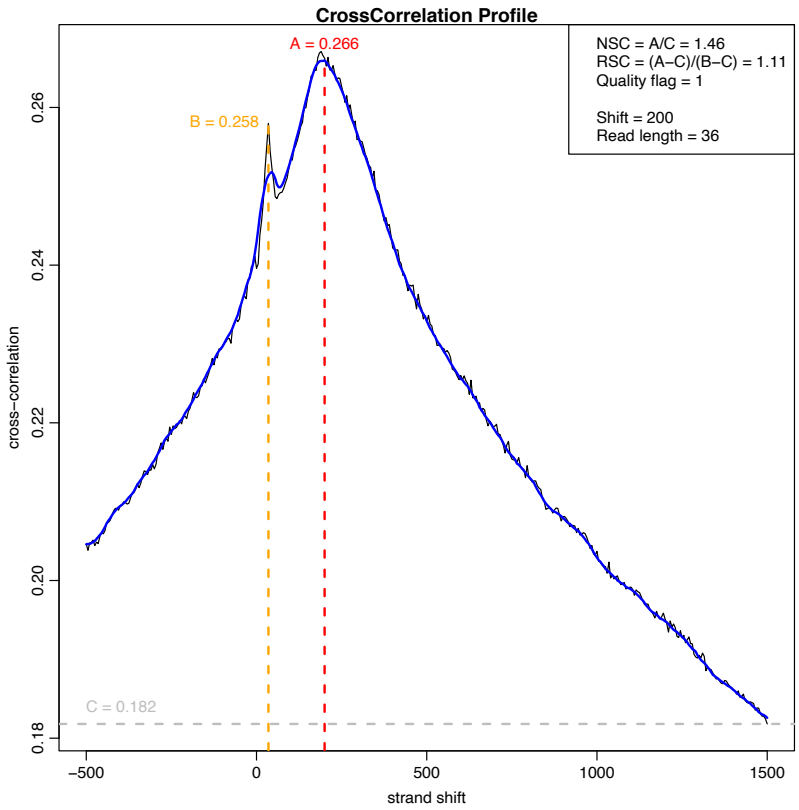

Figure 1: Cross-correlation profile and associated QC metrics as comptued on the subset (chrs 17,18,19) of the test pair of input BAM files.

of the subsequent analyses steps to shift the relative positions of reads mapped on the positive and negative strands (upstream and downstream of the binding sites of the ChIP target).

```
>    finalTagshift<-EM_Results$QCscores_ChIP$tag.shift
```

Another element contained in `"EM_Results"` that will be used in subsequent analysis steps are the reads post filtering to remove local read anomalies. i.e. the ChIP and input control aligned reads are generally filtered as part of ChIP-seq data analyses to filter regions with extremely high read counts compared to the immediate neighborhing positions, which may be due to PCR artifacts. This is done using the spp function to "remove local tag anomalies" (for more details see [5]).

```
> selectedTagsChip<-EM_Results$SelectedTagsChip
> selectedTagsInput<-EM_Results$SelectedTagsInput
```

3.3 Computing Global enrichment profile Metrics (GM)

The second set of quantitative quality control metrics, is based on the global read distribution along the genome for ChIP and Input [3] and we name them Global enrichment profile Metrics (GM). The wrapper function `qualityScores_GM()` reproduces the so-called Fingerprint plot (Figure 2), i.e. the cumulative read distribution plot, from which quantitative QC-metrics are derived as detailed in [1]. Examples of these metrics are the (a) fraction of bins without reads for ChIP and input, (b) the point of maximum distance between the ChIP and input (x-coordinate, y-coordinate for ChIP and input, the distance calculated as absolute difference between the two y-coordinates, the sign of the difference), (c) the fraction of reads in the top 1 percent of bins with highest coverage for ChIP and input.

```
> GM_Results<-qualityScores_GM(
+          selectedTagsChip=selectedTagsChip,
+          selectedTagsInput=selectedTagsInput,
+          tag.shift=finalTagshift,
+          annotationID="hg19")
```

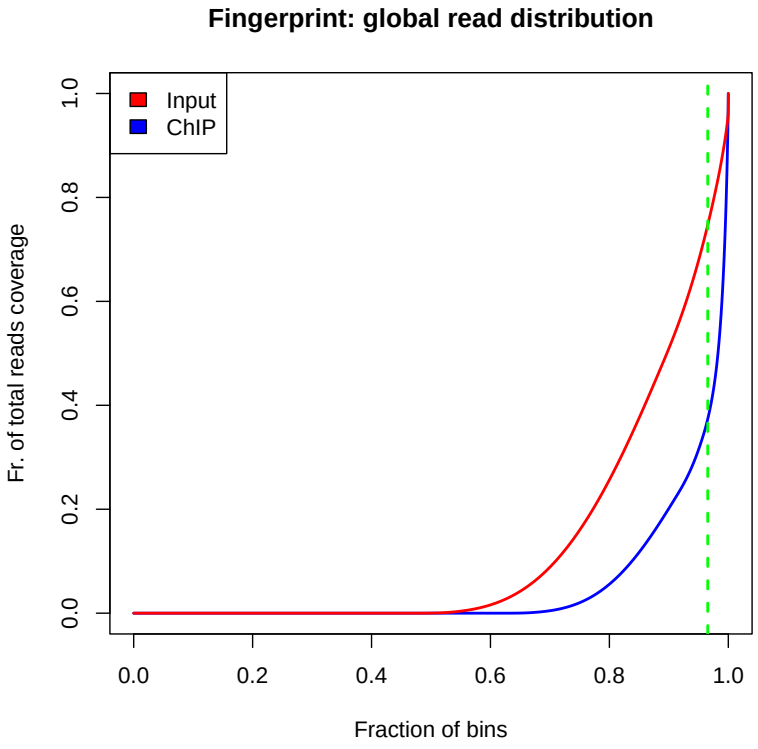

Figure 2: Fingerprint plot of the ChIP sample and its input control.

```
> str(GM_Results)
```

```
List of 9
 $ Ch_X.axis                        : num 0.965
 $ Ch_Y.Input                       : num 0.747
 $ Ch_Y.Chip                        : num 0.374
 $ Ch_sign_chipVSinput              : num 1
 $ Ch_FractionReadsTopbins_chip     : num 0.385
 $ Ch_FractionReadsTopbins_input    : num 0.12
 $ Ch_Fractions_without_reads_chip  : num 0.632
 $ Ch_Fractions_without_reads_input: num 0.485
 $ Ch_DistanceInputChip             : num 0.373
```

3.4 Computing Local enrichment profile metrics (LM)

The third set of quantitative quality control metrics, is based on the local (gene centered) shape of enrichment profiles and we name them Local enrichment profile Metrics (LM) [1]. First we must compute the gene centered profiles of reads distribution (in ChIP and input samples) as well as the normalized ChIP over input enrichment (log2 ratio). The function `createMetageneProfile()` creates the metagene profiles and returns a list with three items: "TSS", "TES" and "geneBody".

```
> Meta_Results<-createMetageneProfile(
+         selectedTagsChip=selectedTagsChip,
+         selectedTagsInput=selectedTagsInput,
+         tag.shift=finalTagshift,
+         annotationID="hg19")
```

The content of the object `"Meta_Result"` is used to create the metagene plots and to extract the LMs for the different profiles.

3.4.1 Plotting metagene profiles

Metagene profiles show the average ChIP-seq reads distribution (for ChIP or control samples separately) or the average ChIP-seq "signal", i.e. the ChIP over input control enrichment (Log2(ChIP/input)) around a set of genomic regions of interest. These may include like the transcription start site (TSS) of annotated genes or their entire gene body.

ChIC creates two types of metagene profiles: (1) the unscaled single-point metagene and (2) the scaled whole gene metagene. In the unscaled single-point metagene, the annotated TSS or annotated transcript end (TES) are used as central point. In the scaled whole gene, the metagene prfiles encompass the entire gene body including 2Kb upstream (promoter) and 1Kb downstream flanking regions.

For the metagene profiles the reads density of the sample is taken over all RefSeq annotated human genes, averaged and log2 transformed. The same is done for the input. The normalised profile is calculated as the signal enrichment (immunoprecipitation over the input) and plotted on the y-axis, whereas the genomic coordinates of the genes like the TSS or regions up- and downstream are shown on the x-axis.

Plotting an unscaled single-point metagene profile The "Meta_Results" object and the `qualityScores_LM()` function can be used to plot the unscaled profile (see Figure 3) and return the respective LM values.

```
> TSSProfile<-qualityScores_LM(
+          data=Meta_Results,
+          tag="TSS", plot="split")
```

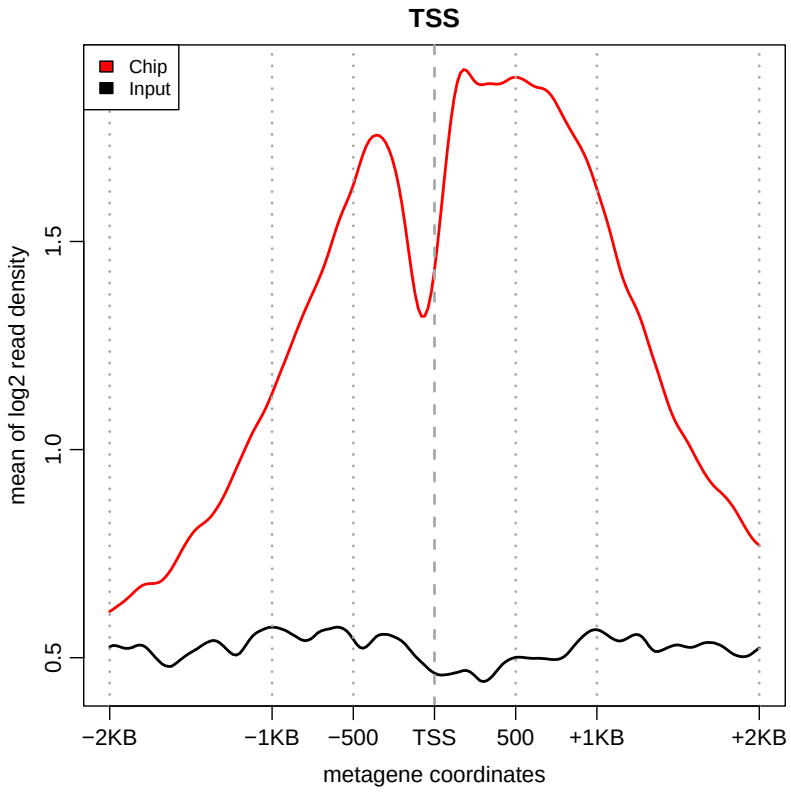

Figure 3: Unscaled single-point metagene profile: reads distribution profile for ChIP and Input at the TSS.

The `"tag"` parameter allows specifying if computing metrics for "TSS", "TES" or "geneBody" regions. The `"plot"` parameter allows to specify if plotting ChIP and input control metaprofiles separately (`plot="split"`), or the normalized ChIP/input signal (`plot="norm"`), or both (`plot="all"` - which is the default

parameter but the output plot should be redirected to a PDF to see both plots on different pages), or none (`plot="none"` - in this latter case only QC metrics are returned but the plots is not drawn). Please note that these output QC metrics are a complex object, not meant to be "human readable", but it is instead meant to be fed to the `predictionScore()` function for computing the ChIC-RF score (see below).

Plotting a scaled whole gene metagene profile The `tag="geneBody"` option is used to plot the scaled metagene profile over the entire length of annotated genes (see Figure 4) and return the respective LM values:

```
> geneBodyProfile<-qualityScores_LM(
+          data=Meta_Results,
+          tag="geneBody", plot="split")
```

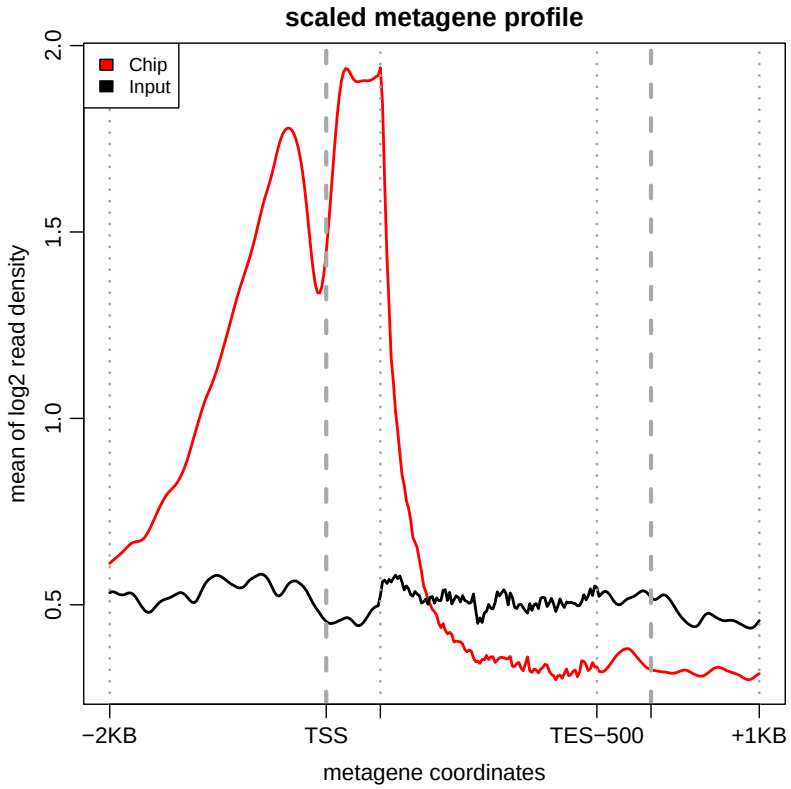

Figure 4: Scaled whole gene metagene profile: reads distribution profile for ChIP and Input along the gene body.

Plotting a scaled whole gene metagene profile, with normalized ChIP/input enrichment signal The `plot="norm"` option is used to plot the normalized

ChIP/input enrichment metagene profile over the entire length of annotated genes (see Figure 5) and return the respective LM values:

```
> geneBodyProfile<-qualityScores_LM(
+          data=Meta_Results,
+          tag="geneBody", plot="norm")
```

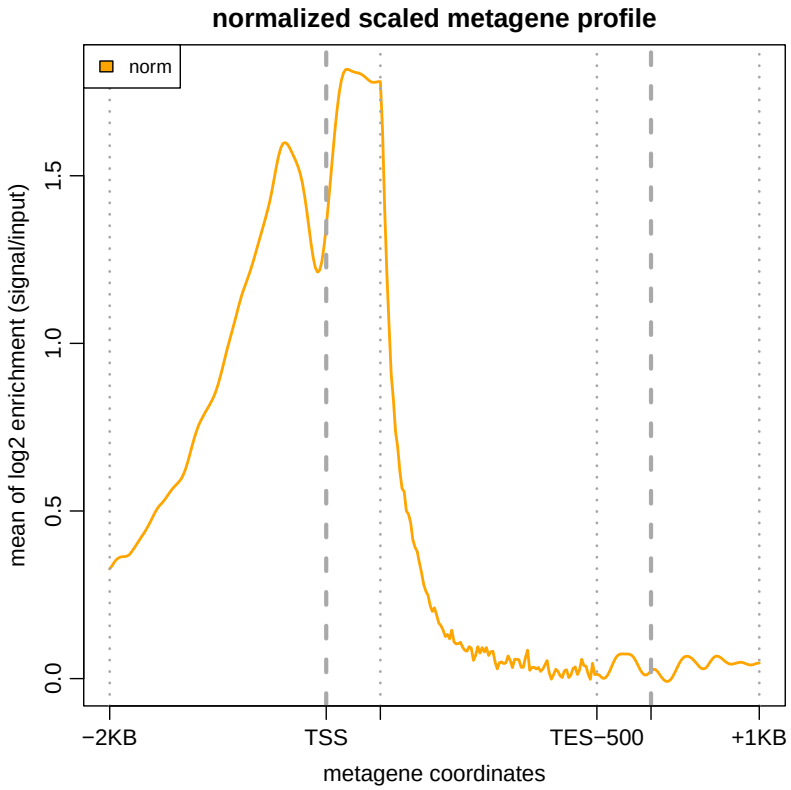

Figure 5: Normalized profile: signal enrichment for ChIP over Input along the gene body for a scaled whole gene profile.

3.5 Comparisons against the reference compendium

The comprehensive set of QC-metrics, computed over a large set of ChIP-seq samples, constitutes in itself a valuable compendium that can be used as a reference for comparison with new samples. ChIC provides three functions for that:

- `metagenePlotsForComparison()` to compare the metagene plots with the compendium
- `plotReferenceDistribution()` to compare a single QC-metric with the compendium values

- `predictionScore()` to obtain a single quality score (ChIC RF score) from the random forest models trained on the previously computed QC-metrics.

List the sample IDs of the compendium To see the IDs of all pre-analyzed ChIP-seq samples from ENCODE and Roadmap that have been included in the compendium by providing the keyword "ENCODE" or "Roadmap".

```
> head(listDatasets(dataset="ENCODE"))
```

```
[1] "ENCFF000AHS" "ENCFF000AHU" "ENCFF000AHZ" "ENCFF000AIB"
[5] "ENCFF000AIE" "ENCFF000AIG"
```

3.5.1 Comparing local enrichment profiles

The `metagenePlotsForComparison()` function can be used to compare the local enrichment profile to the reference compendium by plotting the metagene profile against the expected metagene for the same type of chromatin mark. The expected metagene profile is provided by the compendium mean (black line) and standard error (blue shadow). Examples are shown in Figures 6, 7 and 8.

```
> metagenePlotsForComparison(
+     data = Meta_Results,
+     target = "H3K4me3",
+     tag = "geneBody",
+     plot="chip")
```

The `"plot"` parameter allows to specify if plotting ChIP and input control metaprofiles only (`plot="chip"` or `plot="input"`, respectively), or the normalized ChIP/input signal (`plot="norm"`), or all the three plots (`plot="all"` - which is the default parameter but the output plot should be redirected to a PDF to see each plot on a different page)

```
> metagenePlotsForComparison(
+     data = Meta_Results,
+     target = "H3K4me3",
+     tag = "geneBody",
+     plot="norm")
```

Likewise we can change the `"tag"` parameter to show the profile around TSS or TES, instead than over the `"geneBody"`.

```
> metagenePlotsForComparison(
+     data = Meta_Results,
+     target = "H3K4me3",
+     tag = "TSS",
+     plot="norm")
```

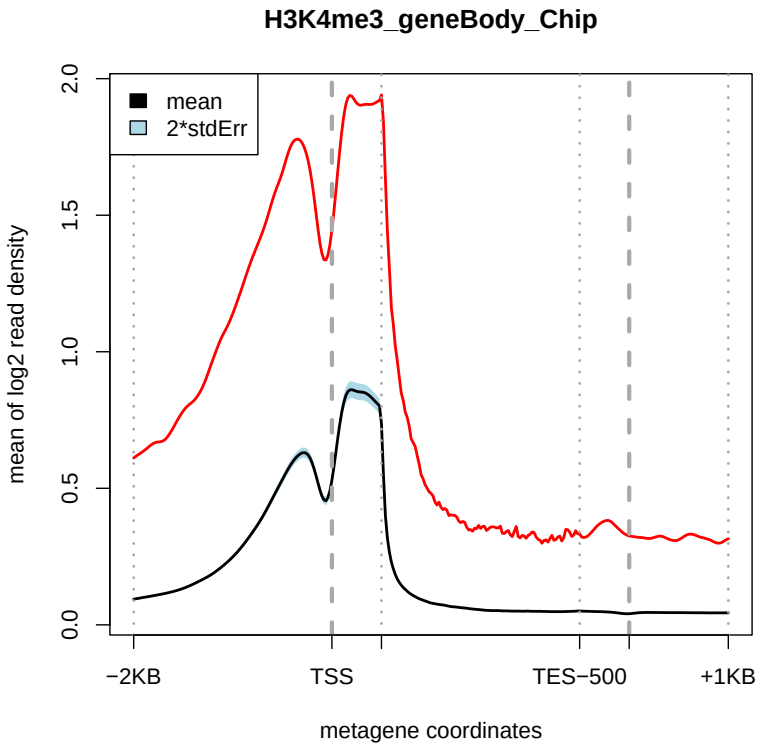

Figure 6: Reads distribution profile plotted against the pre-computed profiles of the compendium. The metagene profile shows the ChIP sample reads distribution (red line) compared to the compendium mean signal (black line) and the 2x standard error (blue shadow).

Please note: In this function the user has to specify the name of the chromatin mark or transcription factor. Please see the `listAvailableElements()` function in the previous paragraphs and sections to get a list of available targets.

Available chromatin marks and transcription factors To visualize the lists of chromatin marks and transcription factors that are available in the compendium for the comparative metagene plots you can use the `listAvailableElements()` as described above. Example:

```
> listAvailableElements(target="mark")
```

```
 [1] "H3K36me3"         "POLR2A"           "H3K4me3"
 [4] "H3K79me2"         "H4K20me1"         "H2AFZ"
 [7] "H3K27me3"         "H3K9me3"          "H3K27ac"
[10] "POLR2AphosphoS5" "H3K9ac"           "H3K4me2"
[13] "H3K9me1"         "H3K4me1"          "POLR2AphosphoS2"
```

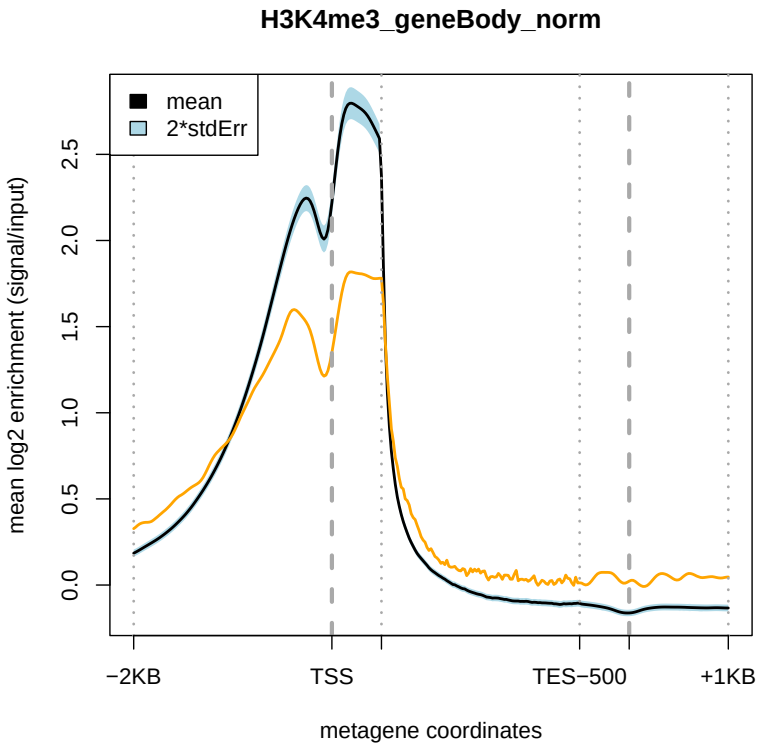

Figure 7: Same as above but plotting the normalized ChIP/input control enrichment

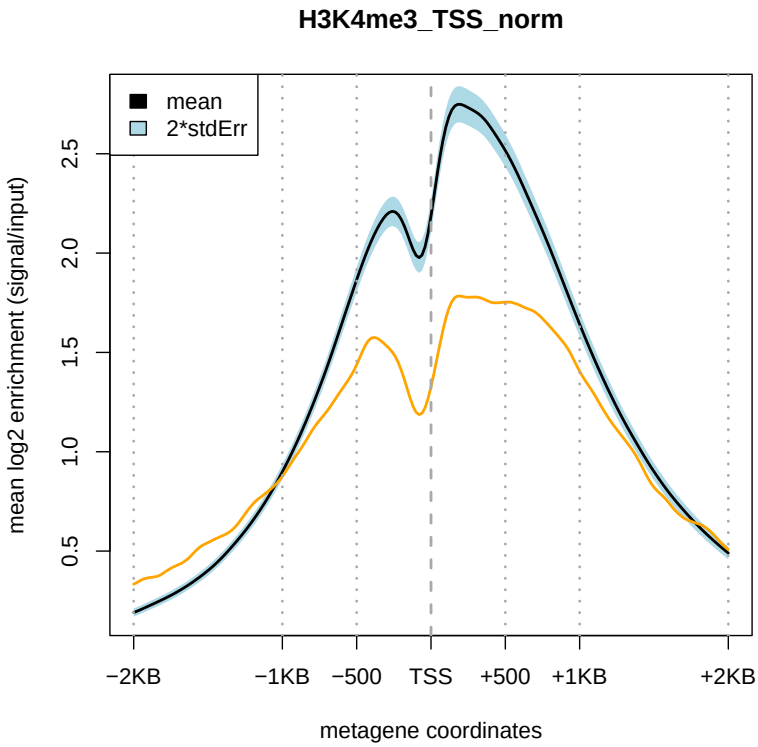

Figure 8: Same as above but plotting the normalized ChIP/input control enrichment around the TSS

```
[16] "H3K79me1"        "H3K4ac"          "H3K14ac"
[19] "H2BK5ac"         "H2BK120ac"       "H2BK15ac"
[22] "H4K91ac"         "H4K8ac"          "H3K18ac"
[25] "H2BK12ac"        "H3K56ac"         "H3K23ac"
[28] "H2AK5ac"         "H2BK20ac"        "H4K5ac"
[31] "H4K12ac"         "H2A.Z"           "H3K23me2"
[34] "H2AK9ac"         "H3T11ph"
```

3.5.2 Comparing QC-metrics to the reference values of the compendium

Plotting a single QC-metric against the reference values from a large number of already published data adds an extra level of information that can be easily accessed. An example is shown in Figure 9.

Please note: To use this function the user has to specify the name of the chromatin mark or transcription factor. Please see `listAvailableElements()` as described above to get a list of available targets.

```
> plotReferenceDistribution(
+     target = "H3K4me3",
+     metricToBePlotted = "RSC",
+     currentValue = EM_Results$QCscores_ChIP$CC_RSC
+     )
```

Available values for `plotReferenceDistribution()` You can use the function `listMetrics()` to visualize the possible values to be passed as parameter `"metricToBePlotted"`. Example:

```
> head(listMetrics("EM"))
```

```
[1] "tag.shift"    "N1"           "Nd"           "StrandShift"
[5] "PBC"          "readLength"
```

3.6 Computing the ChIC RF score

Finally, as discussed in details in the accompanying manuscript [1] the compendium of metrics has been used to train a random forest model that can provide a single score summarizing the sample quality: the ChIC RF score, that can be computed with the `"predictionScore()"` function as in the example below.

```
> ChIC_RFscore <-predictionScore(
+ target="H3K4me3",
+ features_cc=EM_Results,
+ features_global=GM_Results,
```

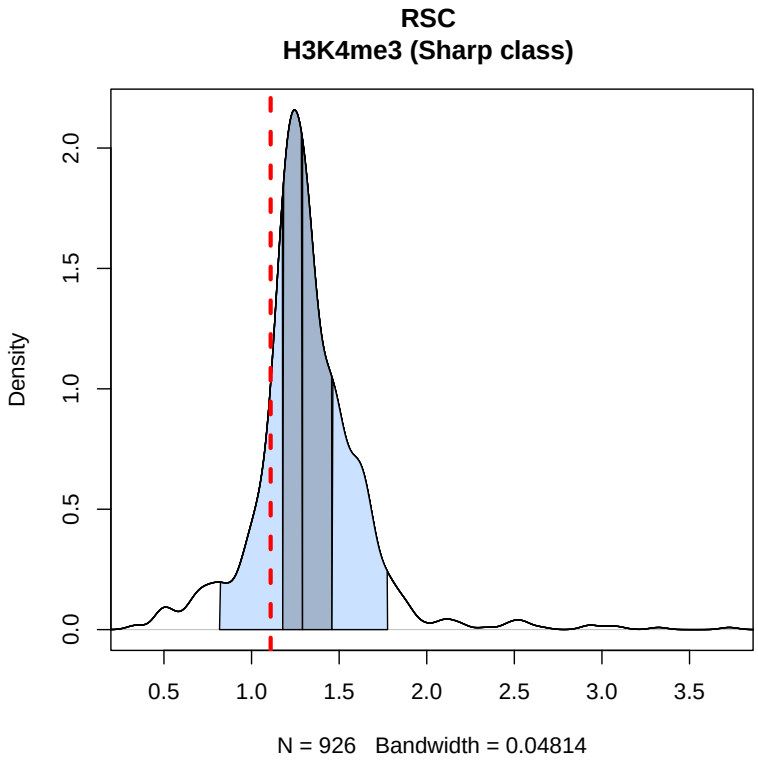

Figure 9: The QC-metric of a newly analysed ChIP-seq sample can be compared to the reference values of the compendium. The density plot shows the QC-metric RSC (red dashed line) of the sample versus the distribution of the same metric in the the compendium for the respective binding profile category (in this case "Sharp class" as indicated in the plot title).

```
+ features_TSS=TSSProfile,
+ features_TES=TESProfile,
+ features_scaled=geneBodyProfile
+ )
```

The final ChIC RF score accounting for the good (positive) quality of the sample is:

```
> ChIC_RFscore["values", "P"]
```

```
[1] 0.5476773
```

References

[1] C. M. Livi, I. Tagliaferri, K. Pal, E. Sebestyén, F. Lucini, A. Bianchi, S. Valsoni, C. Lanzuolo, and F. Ferrari. A ChIC solution for ChIP-seq quality assessment, May 2020. https://www.biorxiv.org/content/10.1101/2020.05.19.103887v1.

[2] S. G. Landt, G. K. Marinov, A. Kundaje, P. Kheradpour, F. Pauli, S. Batzoglou, B. E. Bernstein, P. Bickel, J. B. Brown, P. Cayting, Y. Chen, G. DeSalvo, C. Epstein, K. I. Fisher-Aylor, G. Euskirchen, M. Gerstein, J. Gertz, A. J. Hartemink, M. M. Hoffman, V. R. Iyer, Y. L. Jung, S. Karmakar, M. Kellis, P. V. Kharchenko, Q. Li, T. Liu, X. S. Liu, L. Ma, A. Milosavljevic, R. M. Myers, P. J. Park, M. J. Pazin, M. D. Perry, D. Raha, T. E. Reddy, J. Rozowsky, N. Shoresh, A. Sidow, M. Slattery, J. A. Stamatoyannopoulos, M. Y. Tolstorukov, K. P. White, S. Xi, P. J. Farnham, J. D. Lieb, B. J. Wold, and M. Snyder. ChIP-seq guidelines and practices of the ENCODE and modENCODE consortia. *Genome Res*, 22(9):1813–1831, Sep 2012.

[3] A. Diaz, A. Nellore, and J. S. Song. CHANCE: comprehensive software for quality control and validation of ChIP-seq data. *Genome Biol*, 13(10):R98, Oct 2012.

[4] F. Ramírez, D. P. Ryan, B. Grüning, V. Bhardwaj, F. Kilpert, A. S. Richter, S. Heyne, F. Dündar, and T. Manke. deepTools2: a next generation web server for deep-sequencing data analysis. *Nucleic Acids Res*, 44(W1):W160–165, 07 2016.

[5] P. V. Kharchenko, M. Y. Tolstorukov, and P. J. Park. Design and analysis of ChIP-seq experiments for DNA-binding proteins. *Nat Biotechnol*, 26(12):1351–1359, Dec 2008.

[6] G. K. Marinov, A. Kundaje, P. J. Park, and B. J. Wold. Large-scale quality analysis of published ChIP-seq data. *G3 (Bethesda)*, 4(2):209–223, Feb 2014.

[7] T. S. Carroll, Z. Liang, R. Salama, R. Stark, and I. de Santiago. Impact of artifact removal on ChIP quality metrics in ChIP-seq and ChIP-exo data. *Front Genet*, 5:75, 2014.