

# Package ‘ChIPQC’

April 10, 2023

**Type** Package

**Title** Quality metrics for ChIPseq data

**Version** 1.34.1

**Author** Tom Carroll, Wei Liu, Ines de Santiago, Rory Stark

**Maintainer**

Tom Carroll <tc.infomatics@gmail.com>, Rory Stark <rory.stark@cruk.cam.ac.uk>

**Description** Quality metrics for ChIPseq data.

**biocViews** Sequencing, ChIPSeq, QualityControl, ReportWriting

**License** GPL (>= 3)

**LazyLoad** yes

**Depends** R (>= 3.5.0), ggplot2, DiffBind, GenomicRanges (>= 1.17.19),  
BiocParallel

**Imports** BiocGenerics (>= 0.11.3), S4Vectors (>= 0.1.0), IRanges (>= 1.99.17), Rsamtools (>= 1.17.28), GenomicAlignments (>= 1.1.16), chipseq (>= 1.12.0), gtools, methods, reshape2, Nozzle.R1, Biobase, grDevices, stats, utils, GenomicFeatures, TxDb.Hsapiens.UCSC.hg19.knownGene, TxDb.Hsapiens.UCSC.hg18.knownGene, TxDb.Mmusculus.UCSC.mm10.knownGene, TxDb.Mmusculus.UCSC.mm9.knownGene, TxDb.Rnorvegicus.UCSC.rn4.ensGene, TxDb.Celegans.UCSC.ce6.ensGene, TxDb.Dmelanogaster.UCSC.dm3.ensGene

**Suggests** BiocStyle

**Collate** ChIPQCsample-class.R ChIPQCexperiment-class.R sampleQC.R  
ChIPQC\_IF.R plots.r dbaplots.R utilities.R report.r

**git\_url** <https://git.bioconductor.org/packages/ChIPQC>

**git\_branch** RELEASE\_3\_16

**git\_last\_commit** dabb39a

**git\_last\_commit\_date** 2023-01-10

**Date/Publication** 2023-04-10

**R topics documented:**

ChIPQC-package . . . . .	3
averagepeaksignal-methods . . . . .	4
ChIPQC . . . . .	5
ChIPQC-data . . . . .	8
ChIPQCexperiment-class . . . . .	9
ChIPQCreport-methods . . . . .	11
ChIPQCsample-class . . . . .	12
coveragehistogram-methods . . . . .	15
crosscoverage-methods . . . . .	16
duplicateRate-methods . . . . .	17
duplicates-methods . . . . .	18
flagtagcounts-methods . . . . .	19
fragmentlength-methods . . . . .	19
frip-methods . . . . .	20
mapped-methods . . . . .	21
Normalisedaveragepeaksignal-methods . . . . .	22
peaks-methods . . . . .	23
plotCC-methods . . . . .	23
plotCorHeatmap-methods . . . . .	24
plotCoverageHist-methods . . . . .	25
plotFribl-methods . . . . .	26
plotFrip-methods . . . . .	26
plotPeakProfile-methods . . . . .	27
plotPrincomp-methods . . . . .	28
plotRap-methods . . . . .	29
plotRegi-methods . . . . .	30
plotSSD-methods . . . . .	30
QCannotation-methods . . . . .	31
QCcontrol-methods . . . . .	32
QCdba-methods . . . . .	33
QCmetadata-methods . . . . .	33
QCmetrics-methods . . . . .	34
QCsample-methods . . . . .	35
readlength-methods . . . . .	35
ReadLengthFragmentLengthCrossCoverage-methods . . . . .	36
ReadLengthReadLengthCrossCoverage-methods . . . . .	37
reads-methods . . . . .	37
regi-methods . . . . .	38
RelativeCrossCoverage-methods . . . . .	39
ribl-methods . . . . .	40
rip-methods . . . . .	41
ssd-methods . . . . .	41

**Description**

ChIPQC analyzes aligned reads (in .bam format) for ChIP-seq samples and their associated controls, computing a variety of quality control metrics and statistics, and providing reporting and plotting functions to enable assessment of experimental data for further analysis.

**Details**

Package: ChIPQC  
Type: Package  
Version: 0.1  
Date: 2014-03-01  
License: GPL3

ChIPQC primarily uses two object classes: [ChIPQCsample](#), which encapsulates the information about individual samples, and [ChIPQCexperiment](#), which encapsulates information about larger ChIP-seq experiments (consisting of a number of samples). The primary entry point is the constructor function [ChIPQC](#), which takes a description of an entire experiment, constructs objects for all the samples, and computes the quality metrics.

**Author(s)**

Tom Carroll and Rory Stark

Maintainers: Tom Carroll <tc.infomatics@gmail.com> and Rory Stark <rory.stark@cruk.cam.ac.uk>

**References**

Frontiers?

**See Also**

ChIPQC is designed to work closely with the [DiffBind](#) package, which provides functionality for analyzing ChIP-seq experiments, including performing differential binding analysis to identify significantly differentially bound peaks.

---

averagepeaksignal-methods

*Retrieve average peak profiles*

---

## Description

Retrieve the average peak profile for a sample or set of samples.

## Methods

`signature(object = "ChIPQCexperiment")` Retrieve a matrix of the average peak profiles for all of the samples in a ChIP-seq experiment. Each column represents a sample, and each row a base pair position, centered on peak summits.

`signature(object = "list")` Retrieve a matrix of the average peak profiles for all of the samples in a list of ChIPQC samples. Each column represents a sample, and each row a base pair position, centered on peak summits.

`signature(object = "ChIPQCsample")` Retrieve a vector representing the average peak profile for a sample. Each column represents a basepair position, centered on the peak summits.

## Author(s)

Thomas Carroll and Rory Stark

## See Also

[ChIPQC-package](#), [ChIPQCexperiment](#), [ChIPQCsample](#)

## Examples

```
data(example_QCexperiment)
CTCFprofile = averagepeaksignal(QCsample(exampleExp,1))
length(CTCFprofile)
plot(CTCFprofile,type='l',ylab="mean pileup")

allprofiles = averagepeaksignal(exampleExp)
dim(allprofiles)
for(i in 1:ncol(allprofiles)) lines(allprofiles[,i],col=i)
```

---

ChIPQC                      *Construct a ChIPQCexperiment object*

---

## Description

Constructs a new [ChIPQCexperiment](#) object.

## Usage

```
ChIPQC(experiment, annotation, chromosomes, samples,
        consensus=FALSE, bCount=FALSE, mapQCth=15, blacklist=NULL,
        profileWin=400, fragmentLength=125, shifts=1:300, ...)
```

## Arguments

`experiment`            A specification of the ChIP-seq experiment to evaluate. This can either be a dataframe, a filename for a .csv file, or a [DBA-object](#) object as defined in the [DiffBind](#) package. Columns names in sample sheet may include:

- `SampleID`: Identifier string for sample
- `Tissue`: Identifier string for tissue type
- `Factor`: Identifier string for factor
- `Condition`: Identifier string for condition
- `Treatment`: Identifier string for treatment
- `Replicate`: Replicate number of sample
- `bamReads`: file path for bam file containing aligned reads for ChIP sample
- `bamControl`: file path for bam file containing aligned reads for control sample
- `ControlID`: Identifier string for control sample
- `Peaks`: path for file containing peaks for sample. Format determined by `PeakCaller` field or caller parameter
- `PeakCaller`: Identifier string for peak caller used. If `Peaks` is not a bed file, this will determine how the `Peaks` file is parsed. If missing, will use default peak caller specified in caller parameter. Possible values:
  - “raw”: text file file; peak score is in fourth column
  - “bed”: .bed file; peak score is in fifth column
  - “narrow”: default peak.format: narrowPeaks file
  - “macs”: MACS .xls file
  - “swembl”: SWEMBL .peaks file
  - “bayes”: bayesPeak file
  - “fp4”: FindPeaks v4
- `PeakFormat`: string indicating format for peak files; see `PeakCaller` and [dba.peakset](#)
- `ScoreCol`: column in peak files that contains peak scores

	<ul style="list-style-type: none"> <li>• LowerBetter: logical indicating that lower scores signify better peaks</li> </ul> <p>See the documentation for the <code>sampleSheet</code> parameter of <code>dba</code> for details.</p>
annotation	<p>Either a character string indicating the genome and version to use as a genomic annotation, or a previously defined annotation (obtained using <code>QCannotation</code> on a previously defined <code>ChIPQCexperiment</code> object.) May be left unspecified, in which case no genomic feature analysis is performed. The following annotation specifiers are supported:</p> <ul style="list-style-type: none"> <li>• “hg19”: Human, version 19</li> <li>• “hg18”: Human, version 18</li> <li>• “mm10”: Mouse, version 10</li> <li>• “mm9” : Mouse, version 19</li> <li>• “rn4” : Rat, version 4</li> <li>• “ce6” : C. Elgans, version 6</li> <li>• “dm3” : D. Melanogaster, version 3</li> </ul> <p>Alternatively, you can construct your own annotation; see the package vignette for more information.</p>
chromosomes	<p>Specification of which chromosomes to use for computing QC statistics. If missing, the first chromosome which has a peak is checked. If NULL, all chromosomes will be checked (which may be time-consuming). This can be a character string (e.g. “chr18”) or a vector or list of character strings. If it is an integer or vector of integers, the chromosomes will be checked based on the order that they are listed in a peak set.</p>
samples	<p>list of <code>ChIPCSample</code> objects. If present, the sample objects will be taken directly from this list instead of being computed using the <code>ChIPQCsample</code> constructor.</p>
consensus	<p>If consensus is a <code>GRanges</code> object, all samples will use this peakset when computing peak-based metrics. If <code>consensus=TRUE</code>, a consensus peakset will be generated and used for all samples, derived by merging overlapping peaks in all provided peaksets, keeping any peaks that overlap in at least two samples. To avoid this behavior, set <code>consensus=FALSE</code>; this will result in only supplied peaksets being used for calculation of peak-based metrics (and no peak-based metric being computed for samples with no peakset specified, such as controls).</p>
bCount	<p>if TRUE, the peak scores for all samples will be based on read counts using <code>dba.count</code> using a consensus peakset. If consensus is missing, any samples (such as controls) that are not already associated with a peakset will be associated with the consensus peakset (if consensus is not missing, all samples will be associated with the consensus peakset). Note that the re-counting process may be time-consuming.</p>
mapQcth	<p>An integer representing a mapping quality score threshold. Only reads with mapping quality scores above this threshold will be used for some statistics.</p>
blacklist	<p>A <code>GRanges</code> object or filename specifying a bed file containing genomic regions that should be excluded from the analysis. If missing and the annotation is “hg19”, a default blacklist, <code>blacklist_hg19</code> derived from the UCSC list, will be used. No blacklist is used if this is set to NULL, or is left missing and the annotation is not “hg19”.</p>

<code>profileWin</code>	An integer indicating the width, in base pairs, of the window to be used for peak profiles. Peaks will be centered on their summits, and include half the window size upstream and half downstream of this point.
<code>fragmentLength</code>	An integer indicating the expected fragment length of the libraries. Optional, as this value will be computed for each library.
<code>shifts</code>	A vector of values to try when computing optimal shift sizes.
<code>...</code>	additional parameters passed to <code>dba.count</code> if <code>bCount=TRUE</code> .

### Details

ChIPQC first constructs a new [DBA-object](#) object if one is not provided. Next it computes the annotation if one is not provided. The main loop constructs new [ChIPQCsample](#) objects for each sample (and unique control sample).

### Value

A [ChIPQCexperiment](#) object.

### Note

ChIPQC uses [bplapply](#) from the [BiocParallel](#) package to build the [ChIPQCsample](#) object in parallel, if supported. Control of the parallelization can be effected using [BiocParallel](#) functions, such as [register](#).

### Author(s)

Thomas Carroll and Rory Stark

### See Also

[ChIPQC-package](#), [ChIPQCexperiment](#), [ChIPQCsample](#), [DiffBind](#)

### Examples

```
## Not run: exampleExp = ChIPQC(samples,annotation="hg19")
data(example_QCexperiment)
exampleExp

## Not run: tamoxifen = ChIPQC(samples, ,annotation="hg19", consensus=TRUE, bCounts=T)
data(tamoxifen_QC)
tamoxifen
```

---

ChIPQC-data	<i>Example data sets for ChIPQC package, each containing a ChIPQCexperiment object, as well as a pre-compiled blacklist for hg19.</i>
-------------	---

---

## Description

The tamoxifen\_QC example data set contains a [ChIPQCexperiment](#) object named tamoxifen. This data set, based on Ross-Innes et al (2012), includes 11 ER ChIP libraries, and their input controls, divided into tamoxifen responsive and tamoxifen resistant sample groups. Only data for chromosome 18 (chr18) are included.

The example\_QCexperiment data set contains a [ChIPQCexperiment](#) object named exampleExp. This data set, derived from ENCODE data, includes 6 ChIP libraries. Only data for chromosome 22 (chr22) are included.

blacklist\_hg19 data set includes a [GRanges](#) object named blacklist\_hg19 containing blacklisted regions for the human genome, derived from the UCSC blacklist.

## Usage

```
data(tamoxifen_QC)
data(example_QCexperiment)
data(blacklist_hg19)
```

## Format

tamoxifen\_QC: A single [ChIPQCexperiment](#) object named tamoxifen is loaded. This object is used for the [ChIPQC-package](#) examples and vignette. This object can also be used with the [DiffBind](#) package (see related data objects [tamoxifen](#)).

example\_QCexperiment: A single [ChIPQCexperiment](#) object named exampleExp is loaded. This object is used for the [ChIPQC-package](#) examples and vignette.

blacklist\_hg19: A single [GRanges](#) object named blacklist.hg19 that is used by default when processing hg19 data sets.

## Source

Ross-Innes, C. S., Stark, R., Teschendorff, A. E., Holmes, K. A., Ali, H. R., Dunning, M. J., Brown, G. D., Gojis, O., Ellis, I. O., Green, A. R., Ali, S., Chin, S.-F., Palmieri, C., Caldas, C., and Carroll, J. S. (2012). Differential oestrogen receptor binding is associated with clinical outcome in breast cancer. *Nature* 481, 389-393.

## Examples

```
data(tamoxifen_QC)
tamoxifen
plotRegi(tamoxifen)
```



```
data(example_QCexperiment)
exampleExp
```

```
data(blacklist_hg19)
blacklist.hg19
```

---

ChIPQCexperiment-class

*ChIPQCexperiment instances*

---

## Description

The ChIPQCexperiment class is built around a list of [ChIPQCsample](#) objects, each representing a ChIP or control sample in a ChIP-seq experiment. These objects are created using the [ChIPQC](#) function.

## Slots

`.Data`: Object of class "list": internal

`Samples`: Object of class "list": List of [ChIPQCsample](#) objects.

`DBA`: Object of class "DBA": [DBA-object](#) object (from package [DiffBind](#))

`annotation`: Object of class "list" : annotation data

## Extends

Class "[list](#)".

## Methods

**QCmetadata** signature(object = "ChIPQCexperiment"): see [QCmetadata](#).

**QCmetrics** signature(object = "ChIPQCexperiment"): see [QCmetrics](#).

**QCsample** signature(object = "ChIPQCexperiment"): see [QCsample](#).

**QCcontrol** signature(object = "ChIPQCexperiment"): see [QCcontrol](#).

**QCannotation** signature(object = "ChIPQCexperiment"): see [QCannotation](#).

**QCdba** signature(object = "ChIPQCexperiment"): see [QCdba](#).

**averagepeaksignal** signature(object = "ChIPQCexperiment"): see [averagepeaksignal](#).

**coveragehistogram** signature(object = "ChIPQCexperiment"): see [coveragehistogram](#).

**crosscoverage** signature(object = "ChIPQCexperiment"): see [crosscoverage](#).

**flagtagcounts** signature(object = "ChIPQCexperiment"): see [flagtagcounts](#).

**fragmentlength** signature(object = "ChIPQCexperiment"): see [fragmentlength](#).

**FragmentLengthCrossCoverage** signature(object = "ChIPQCexperiment"): see [FragmentLengthCrossCoverage](#).

**frip** signature(object = "ChIPQCexperiment"): see [frip](#).

**mapped** signature(object = "ChIPQCexperiment"): see [mapped](#).

**reads** signature(object = "ChIPQCexperiment"): see [reads](#).

**duplicates** signature(object = "ChIPQCexperiment"): see [duplicates](#).

**duplicateRate** signature(object = "ChIPQCexperiment"): see [duplicateRate](#).

**Normalisedaveragepeaksignal** signature(object = "ChIPQCexperiment"): see [Normalisedaveragepeaksignal](#).

**peaks** signature(object = "ChIPQCexperiment"): see [peaks](#).

**readlength** signature(object = "ChIPQCexperiment"): see [readlength](#).

**ReadLengthCrossCoverage** signature(object = "ChIPQCexperiment"): see [ReadLengthCrossCoverage](#).

**RelativeCrossCoverage** signature(object = "ChIPQCexperiment"): see [RelativeCrossCoverage](#).

**rib1** signature(object = "ChIPQCexperiment"): see [rib1](#).

**rip** signature(object = "ChIPQCexperiment"): see [rip](#).

**show** signature(object = "ChIPQCexperiment"): see [show](#).

**ssd** signature(object = "ChIPQCexperiment"): see [ssd](#).

**regi** signature(object = "ChIPQCexperiment"): see [regi](#).

**plotCC** signature(object = "ChIPQCexperiment"): see [plotCC](#).

**plotCoverageHist** signature(object = "ChIPQCexperiment"): see [plotCoverageHist](#).

**plotFrib1** signature(object = "ChIPQCexperiment"): see [plotFrib1](#).

**plotPeakProfile** signature(object = "ChIPQCexperiment"): see [plotPeakProfile](#).

**plotRap** signature(object = "ChIPQCexperiment"): see [plotRap](#).

**plotRegi** signature(object = "ChIPQCexperiment"): see [plotRegi](#).

**plotCorHeatmap** signature(object = "ChIPQCexperiment"): see [plotCorHeatmap](#).

**plotPrincomp** signature(object = "ChIPQCexperiment"): see [plotPrincomp](#).

**ChIPQCreport** signature(object = "ChIPQCexperiment"): see [ChIPQCreport](#).

**Author(s)**

Thomas Carroll and Rory Stark

**See Also**

[ChIPQC-package](#), [ChIPQCsample](#), [DiffBind](#)

**Examples**

```
## Not run: exampleExp = ChIPQC(samples)
data(example_QCexperiment)
exampleExp

## Not run: tamoxifen = ChIPQC(samples, consensus=TRUE, bCounts=T)
data(tamoxifen_QC)
tamoxifen
```

---

ChIPQCreport-methods *Generate a summary QC report*

---

## Description

Generate an HTML summary QC report.

## Methods

`signature(experiment = "ChIPQCexperiment", facet=TRUE, reportName="ChIPQC", reportFolder="ChIPQCreport")`  
 Generates an summary QC report for the experiment in HTML format.

experiment	<a href="#">ChIPQCexperiment</a> object
facet	logical indicating whether or not to facet using experimental metadata.
reportName	filename of main report file (.html).
reportFolder	directory name where plot graphics will be saved
facetBy	metadata fields to use for faceting
colourBy	metadata field to color by

`signature(experiment = "list", facet=TRUE, reportName="ChIPQC", reportFolder="ChIPQCreport", facetBy=c)`  
 Generates an summary QC report for a list of ChIPQCsample objects in HTML format.

experiment	<a href="#">list</a> object
facet	logical indicating whether or not to facet using experimental metadata.
reportName	filename of main report file (.html).
reportFolder	directory name where plot graphics will be saved
facetBy	metadata fields to use for faceting
colourBy	metadata field to color by

`signature(sample = "ChIPQCsample", reportName="ChIPQC", reportFolder="ChIPQCreport",)`  
 Generate a summary QC report for a sample in HTML format.

sample	<a href="#">ChIPQCsample</a> object
reportName	filename of main report file (.html).
reportFolder	directory name where plot graphics will be saved

## Note

ChIPQCreport uses Nozzle.R2 for generating HTML.

## Author(s)

Thomas Carroll and Rory Stark

**See Also**

[ChIPQC-package](#), [ChIPQCexperiment](#), [ChIPQCsample](#)

**Examples**

```
data(example_QCexperiment)
ChIPQCreport(exampleExp, facetBy=c("Tissue", "Factor"))
#report in ChIPQCreport/Example.html

data(tamoxifen_QC)
ChIPQCreport(tamoxifen, facetBy="Tissue", colourBy="Condition")
#report in ChIPQCreport/ChIPQC.html

ChIPQCreport(tamoxifen, facetBy=c("Tissue", "Condition"))
#report in ChIPQCreport/ChIPQC.html
```

---

ChIPQCsample-class      *Class "ChIPQCsample"*

---

**Description**

Object containing quality metrics computed for a ChIP-seq (or associated control) sample.

**Objects from the Class**

Objects can be created using the [ChIPQCsample](#) function.

**Constructor Function**

`ChIPQCsample(reads, peaks, annotation, chromosomes = NULL, mapQCth = 15, blacklist, profileWin = 400, fragmentLength = 125, shifts = 1:300, runCrossCor = FALSE, verboseT=FALSE)`

- `readscharacter` string filename of .bam file
- `peaksGRanges` object or character string filename of peaks. If present, peak-based metrics will be computed.
- `annotation` Either a character string indicating the genome and version to use as a genomic annotation, or a previously defined annotation (obtained using [QCannotation](#) on a previously defined [ChIPQCexperiment](#) object.) May be left unspecified, in which case no genomic feature analysis is performed. The following annotation specifiers are supported:

"hg20"	Human, version 20
"hg19"	Human, version 19
"hg18"	Human, version 18
"mm10"	Mouse, version 10
"mm9"	Mouse, version 19
"rn4"	Rat, version 4
"ce6"	C. Elgans, version 6
"dm3"	D. Melanogaster, version 3

Alternatively, you can construct your own annotation; see the package Vignette for more information.

- **chromosomes** Specification of which chromosomes to use for computing QC statistics. If missing, the first chromosome which has a peak is checked. If NULL, all chromosomes will be checked (which may be time-consuming). This can be a character string (e.g. "chr18") or a vector or list of character strings. If it is an integer or vector of integers, the chromosomes will be checked based on the order that they are listed in a peak set.
- **mapQCth** An integer representing a mapping quality score threshold. Only reads with mapping quality scores above this threshold will be used for some statistics.
- **blacklist** A [GRanges](#) object or filename specifying a bed file containing genomic regions that should be excluded from the analysis. If missing and the annotation is "hg19", a default blacklist, [blacklist\\_hg19](#) derived from the UCSC list, will be used. No blacklist is used if this is set to NULL, or is left missing and the annotation is not "hg19".
- **profileWin** An integer indicating the width, in base pairs, of the window to be used for peak profiles. Peaks will be centered on their summits, and include half the window size upstream and half downstream of this point.
- **fragmentLength** An integer indicating the expected fragment length of the libraries. Optional, as this value will be computed.
- **shifts** A vector of values to try when computing optimal shift sizes.
- **runCrossCor** Compute cross-correlation in addition to cross-coverage. This will take more compute time, and is currently not used in the final report.
- **verboseT** TRUE or FALSE, specifying whether to report progress. Default is TRUE. When set to FALSE ChIPQC does not report any progress until complete.

## Slots

**AveragePeakSignal:** Object of class "list"  
**CrossCoverage:** Object of class "numeric"  
**CrossCorrelation:** Object of class "numeric"  
**SSD:** Object of class "numeric"  
**SSDBL:** Object of class "numeric"  
**CountsInPeaks:** Object of class "numeric"  
**CountsInBlackList:** Object of class "numeric"  
**CountsInFeatures:** Object of class "list"  
**PropInFeatures:** Object of class "list"  
**CoverageHistogram:** Object of class "numeric"  
**FlagAndTagCounts:** Object of class "numeric"  
**readlength:** Object of class "numeric"  
**seqnames:** Object of class "Rle"  
**ranges:** Object of class "IRanges"  
**strand:** Object of class "Rle"  
**elementMetadata:** Object of class "DataFrame"  
**seqinfo:** Object of class "Seqinfo"  
**metadata:** Object of class "list"

**Extends**

Class "[GRanges](#)"

**Methods**

**averagepeaksignal** signature(object = "ChIPQCsample"): see [averagepeaksignal](#).  
**coveragehistogram** signature(object = "ChIPQCsample"): see [coveragehistogram](#).  
**crosscoverage** signature(object = "ChIPQCsample"): see [crosscoverage](#).  
**flagtagcounts** signature(object = "ChIPQCsample"): see [flagtagcounts](#).  
**fragmentlength** signature(object = "ChIPQCsample"): see [fragmentlength](#).  
**FragmentLengthCrossCoverage** signature(object = "ChIPQCsample"): see [FragmentLengthCrossCoverage](#).  
**frip** signature(object = "ChIPQCsample"): see [frip](#).  
**mapped** signature(object = "ChIPQCsample"): see [mapped](#).  
**reads** signature(object = "ChIPQCsample"): see [reads](#).  
**duplicates** signature(object = "ChIPQCsample"): see [duplicates](#).  
**duplicateRate** signature(object = "ChIPQCsample"): see [duplicateRate](#).  
**Normalisedaveragepeaksignal** signature(object = "ChIPQCsample"): see [Normalisedaveragepeaksignal](#).  
**peaks** signature(object = "ChIPQCsample"):see [peaks](#).  
**readlength** signature(object = "ChIPQCsample"): see [readlength](#).  
**ReadLengthCrossCoverage** signature(object = "ChIPQCsample"): see [ReadLengthCrossCoverage](#).  
**RelativeCrossCoverage** signature(object = "ChIPQCsample"):see [RelativeCrossCoverage](#).  
**ribl** signature(object = "ChIPQCsample"): see [ribl](#).  
**rip** signature(object = "ChIPQCsample"): see [rip](#).  
**show** signature(object = "ChIPQCsample"): see [show](#).  
**ssd** signature(object = "ChIPQCsample"): see [ssd](#).  
**regi** signature(object = "ChIPQCsample"): see [regi](#).  
**plotCC** signature(object = "ChIPQCsample"): see [plotCC](#).  
**plotCoverageHist** signature(object = "ChIPQCsample"): see [plotCoverageHist](#).  
**plotFribl** signature(object = "ChIPQCsample"): see [plotFribl](#).  
**plotPeakProfile** signature(object = "ChIPQCsample"): see [plotPeakProfile](#).  
**plotRap** signature(object = "ChIPQCsample"): see [plotRap](#).  
**plotRegi** signature(object = "ChIPQCsample"): see [plotRegi](#).

**Author(s)**

Thomas Carroll and Rory Stark

**References**

Carroll TS, Liang Z, Salama R, Stark R and Santiago Id (in press). Impact of artefact removal on ChIP quality metrics in ChIP-seq and ChIP-exo data. *Frontiers in Genetics*.

**See Also**

[ChIPQC-package](#), [ChIPQCsample](#)

**Examples**

```
bamFile <- system.file("extdata", "ex1.bam",  
                      package="Rsamtools")  
ex1 <- ChIPQCsample(bamFile,annotation=NULL)  
readlength(ex1)  
fragmentlength(ex1)
```

---

coveragehistogram-methods

*Retrieve histogram data representing densities of coverage pileups*

---

**Description**

Retrieve histogram data representing densities of coverage pileups.

**Methods**

`signature(object = "ChIPQCexperiment")` Retrieve a matrix of coverage histogram data for all samples in a ChIP-seq experiment. Each column represents a sample, and each row a pileup height, with the value representing the number of basepair positions that report this pileup height.

`signature(object = "list")` Retrieve a matrix of coverage histogram data for all ChIPQCsamples in a list. Each column represents a sample, and each row a pileup height, with the value representing the number of basepair positions that report this pileup height.

`signature(object = "ChIPQCsample")` Retrieve a vector representing coverage histogram data for a sample. Values represent the number of base pairs positions that report the pileup value. The value in position 1 of the vector contains the number of examined basepair positions that are overlapped by exactly zero reads, while position 2 shows the number of basepair positions overlapped by exactly one read, etc.

**Author(s)**

Thomas Carroll and Rory Stark

**See Also**

[ChIPQC-package](#), [ChIPQCexperiment](#), [ChIPQCsample](#)

**Examples**

```

data(example_QCexperiment)
CTCFcoverage = coveragehistogram(QCsample(exampleExp,1))
length(CTCFcoverage)
plot(log10(CTCFcoverage),type='l',ylab="log10 Density",
      xlab="Pileup")

allcoverages = coveragehistogram(exampleExp)
dim(allcoverages)
for(i in 1:ncol(allcoverages)) lines(log10(allcoverages[,i]),col=i)

```

---

crosscoverage-methods *Retrieve the cross coverage values for a range of shift sizes*

---

**Description**

Retrieves the cross-coverage values for a range of shift sizes.

**Methods**

`signature(object = "ChIPQCexperiment")` Retrieve a matrix of cross-coverage data for all samples in an ChIP-seq experiment. Each column represents a sample, and each row a shift size, with the value representing the cross-coverage using that size read.

`signature(object = "list")` Retrieve a matrix of cross-coverage data for all samples in a list of ChIPQCsample objects. Each column represents a sample, and each row a shift size, with the value representing the cross-coverage using that size read.

`signature(object = "ChIPQCsample")` Retrieve a vector of cross-coverage data for a sample. Each position in the vector corresponds to a shift size, with the value representing the cross-coverage using that size read.

**Author(s)**

Thomas Carroll and Rory Stark

**See Also**

[ChIPQC-package](#), [ChIPQCexperiment](#), [ChIPQCsample](#)

**Examples**

```

data(example_QCexperiment)
CTCFcoverage = crosscoverage(QCsample(exampleExp,1))
length(CTCFcoverage)
plot(CTCFcoverage,type='l',
      ylab="Cross-coverage",
      xlab="Fragment length")

allcoverages = crosscoverage(exampleExp)

```



```
dim(allcoverages)
for(i in 1:ncol(allcoverages)) lines(allcoverages[,i],col=i)
```

---

duplicateRate-methods *Retrieve duplication rates*

---

## Description

Retrieve duplication rates.

## Methods

`signature(object = "ChIPQCexperiment", bFiltered)` Retrieve a vector of the duplication rates for each sample in an experiment. A read is considered duplicated if another read maps to the same genomic location; the duplication rate is the number of duplicated reads divided by the total number of reads for a sample. If `bFiltered=TRUE` (or is missing), only reads that pass the mapping quality filter for each sample are included. if `bFiltered=FALSE`, all reads for each sample will be included.

`signature(object = "list", bFiltered)` Retrieve a vector of the duplication rates for each sample in a list of `ChIPQCsample` objects. A read is considered duplicated if another read maps to the same genomic location; the duplication rate is the number of duplicated reads divided by the total number of reads for a sample. If `bFiltered=TRUE` (or is missing), only reads that pass the mapping quality filter for each sample are included. if `bFiltered=FALSE`, all reads for each sample will be included.

`signature(object = "ChIPQCsample", bFiltered)` Retrieve the duplication rate for a sample. A read is considered duplicated if another read maps to the same genomic location; the duplication rate is the number of duplicated reads divided by the total number of reads for the sample. If `bFiltered=TRUE` (or is missing), only reads that pass the mapping quality filter for the sample are included. if `bFiltered=FALSE`, all reads for the sample will be included.

## Author(s)

Thomas Carroll and Rory Stark

## See Also

[ChIPQC-package](#), [ChIPQCexperiment](#), [ChIPQCsample](#)

## Examples

```
data(example_QCexperiment)
duplicateRate(exampleExp)
duplicateRate(QCsample(exampleExp,1))
```

---

duplicates-methods      *Retrieve numbers of duplicate reads.*

---

## Description

Retrieve the numbers of duplicate reads.

## Methods

`signature(object = "ChIPQCexperiment", bFiltered)` Retrieve a vector of the numbers of duplicate reads for each sample in an experiment. A read is considered duplicated if another read maps to the same genomic location. If `bFiltered=TRUE` (or is missing), this will be the number of duplicates that pass the mapping quality filter for each sample. if `bFiltered=FALSE`, it will be the total number of duplicates for each sample.

`signature(object = "list", bFiltered)` Retrieve a vector of the numbers of duplicate reads for each sample in a list of `ChIPQCsample` objects. A read is considered duplicated if another read maps to the same genomic location. If `bFiltered=TRUE` (or is missing), this will be the number of duplicates that pass the mapping quality filter for each sample. if `bFiltered=FALSE`, it will be the total number of duplicates for each sample.

`signature(object = "ChIPQCsample", bFiltered)` Retrieve the number of duplicates for a sample. A read is considered duplicated if another read maps to the same genomic location. If `bFiltered=TRUE` (or is missing), this will be the number of duplicates that pass the mapping quality filter. if `bFiltered=FALSE`, it will be the total number of duplicates for the sample.

## Author(s)

Thomas Carroll and Rory Stark

## See Also

[ChIPQC-package](#), [ChIPQCexperiment](#), [ChIPQCsample](#)

## Examples

```
data(example_QCexperiment)
duplicates(exampleExp)
duplicates(QCsample(exampleExp,1))
```

---

flagtagcounts-methods *Retrieve numbers of reads that pass various filters*

---

### Description

Retrieve numbers of reads that pass various filters

### Methods

signature(object = "ChIPQCexperiment") Retrieve a matrix of counts passing various filters for all the samples in an experiment. Each column represents the counts in a sample.

signature(object = "list") Retrieve a matrix of counts passing various filters for all the samples in a list of ChIPQCsample objects. Each column represents the counts in a sample.

signature(object = "ChIPQCsample") Retrieve a vector of counts passing various filters. The values are:

UnMapped	Number of reads that are not mapped (aligned)
Mapped	Number of reads that are mapped (aligned)
Duplicates	Number of reads that align to exactly the same place as another read
MapQPass	Number of reads with a mapping quality score greater than or equal to the specified threshold
MapQPassandDup	Number of reads that are mapped (aligned) and not duplicates

### Author(s)

Thomas Carroll and Rory Stark

### See Also

[ChIPQC-package](#), [ChIPQCexperiment](#), [ChIPQCsample](#)

### Examples

```
data(example_QCexperiment)
flagtagcounts(exampleExp)
flagtagcounts(QCsample(exampleExp,1))
```

---

fragmentlength-methods

*Retrieve the estimated fragment length*

---

### Description

Retrieve the estimated fragment length.

**Methods**

`signature(object = "ChIPQCexperiment")` Retrieve a vector of estimated fragments sizes, one for each sample in the experiment.

`signature(object = "list")` Retrieve a vector of estimated fragments sizes, one for each sample in a list of ChIPQCsample objects.

`signature(object = "ChIPQCsample", width)` Retrieve the estimated fragment length for a sample. If width is missing, the readlength derived from the bam file is used as the read length.

**Author(s)**

Thomas Carroll and Rory Stark

**See Also**

[ChIPQC-package](#), [ChIPQCexperiment](#), [ChIPQCsample](#)

**Examples**

```
data(example_QCexperiment)
fragmentlength(exampleExp)
fragmentlength(QCsample(exampleExp,1))
```

---

frip-methods

*Retrieve fraction of reads in peaks*

---

**Description**

Retrieve the fraction of reads in peaks

**Methods**

`signature(object = "ChIPQCexperiment")` Retrieve a vector of values representing the proportion of reads that overlap peaks for each sample in an experiment.

`signature(object = "list")` Retrieve a vector of values representing the proportion of reads that overlap peaks for each sample in a list of ChIPQCsample objects.

`signature(object = "ChIPQCsample")` Retrieve a value representing the proportion of reads that overlap the peaks for a sample.

**Author(s)**

Thomas Carroll and Rory Stark

**See Also**

[ChIPQC-package](#), [ChIPQCexperiment](#), [ChIPQCsample](#)

**Examples**

```
data(example_QCexperiment)
frip(exampleExp)
frip(QCsample(exampleExp,1))
```

---

mapped-methods

*Retrieve numbers of mapped reads*

---

**Description**

Retrieve the numbers of mapped reads.

**Methods**

signature(object = "ChIPQCexperiment") Retrieve a vector of the numbers of mapped (aligned) reads for each sample in an experiment.

signature(object = "list") Retrieve a vector of the numbers of mapped (aligned) reads for each sample in a list of ChIPQCsample objects.

signature(object = "ChIPQCsample") Retrieve the number of mapped (aligned) reads in a sample.

**Author(s)**

Thomas Carroll and Rory Stark

**See Also**

[ChIPQC-package](#), [ChIPQCexperiment](#), [ChIPQCsample](#)

**Examples**

```
data(example_QCexperiment)
mapped(exampleExp)
mapped(QCsample(exampleExp,1))
```

---

Normalisedaveragepeaksignal-methods

*Retrieve normalised average peak profiles*

---

## Description

Retrieve normalised average peak profiles

## Methods

`signature(object = "ChIPQCexperiment")` Retrieve a matrix of normalised average peak signal data for all samples in a ChIP-seq experiment. Each column represents a sample, and each row a base pair position, centered on peak summits.

`signature(object = "list")` Retrieve a matrix of normalised average peak signal data for all samples in a list of ChIPQCsample objects. Each column represents a sample, and each row a base pair position, centered on peak summits.

`signature(object = "ChIPQCsample")` Retrieve a vector representing the normalised average peak profile for a sample. Each column represents a basepair position, centered on the peak summits.

## Author(s)

Thomas Carroll and Rory Stark

## See Also

[ChIPQC-package](#), [ChIPQCexperiment](#), [ChIPQCsample](#)

## Examples

```
data(example_QCexperiment)
CTCFprofile = Normalisedaveragepeaksignal(QCsample(exampleExp,1))
length(CTCFprofile)
plot(CTCFprofile,type='l',ylab="normalised mean pileup")

allprofiles = Normalisedaveragepeaksignal(exampleExp)
dim(allprofiles)
for(i in 1:ncol(allprofiles)) lines(allprofiles[,i],col=i)
```

---

peaks-methods      *Retrieve peaks*

---

**Description**

Retrieve peaks.

**Methods**

signature(object = "ChIPQCexperiment") Retrieve a [GRangesList](#) of the peaks associated with all the samples in an experiment.

signature(object = "list") Retrieve a [GRangesList](#) of the peaks associated with all the samples in a list of ChIPQCsample objects.

signature(object = "ChIPQCsample") Retrieve a [GRanges](#) object containing the peaks associated with a sample.

**Author(s)**

Thomas Carroll and Rory Stark

**See Also**

[ChIPQC-package](#), [ChIPQCexperiment](#), [ChIPQCsample](#)

**Examples**

```
data(example_QCexperiment)
expPeaks = peaks(exampleExp)
length(expPeaks)
peaks(QCsample(exampleExp,1))
```

---

plotCC-methods      *Generate Cross-Coverage plots*

---

**Description**

Generate Cross-Coverage plots.

**Methods**

signature(object = "ChIPQCexperiment", method) Generate cross-coverage plots for all the samples in an experiment.

signature(object = "list", method) Generate cross-coverage plots for list of samples in an experiment.

signature(object = "ChIPQCsample", methods) Generate cross-coverage plots for a sample. Supported methods include:

"Coverage" [default] Coverage plot

**Note**

plotCC uses ggplot2 for plotting, and returns a ggplot2 plot dataframe.

**Author(s)**

Thomas Carroll and Rory Stark

**See Also**

[ChIPQC-package](#), [ChIPQCexperiment](#), [ChIPQCsample](#)

**Examples**

```
data(example_QCexperiment)
plotCC(exampleExp)
plotCC(exampleExp,excludedBox=TRUE)
plotCC(QCsample(exampleExp,1))
plotCC(QCsample(exampleExp)[1:4])
```

---

plotCorHeatmap-methods

*Generate Correlation Heatmap for ChIP samples*

---

**Description**

Generates correlation heatmap for ChIP samples.

**Methods**

signature(object = "ChIPQCexperiment", attributes, ...) Generate correlation heatmap, including clustering dendrogram, for all the samples in an experiment that are associated with a peakset.

attributes character string, or vector of character strings, representing metadata field names, for use in labeling  
... additional parameters passed to [dba.plotHeatmap](#)

**Note**

plotCorHeatmap uses [dba.plotHeatmap](#) for plotting.

**Author(s)**

Rory Stark and Thomas Carroll



**See Also**

[ChIPQC-package](#), [ChIPQCexperiment](#), [dba.plotHeatmap](#)

**Examples**

```
data(tamoxifen_QC)
plotCorHeatmap(tamoxifen, attributes=c("Tissue", "Condition", "Replicate"))
```

---

plotCoverageHist-methods

*Generate coverage histogram plot*

---

**Description**

Generate coverage histogram plot.

**Methods**

signature(object = "ChIPQCexperiment") Generate coverage histogram plots for all the samples in an experiment.

signature(object = "list") Generate coverage histogram plots for all the samples in a list of ChIPQCsamples.

signature(object = "ChIPQCsample") Generate coverage histogram plots for a sample.

**Note**

Uses ggplot2 for plotting, and returns a ggplot2 plot dataframe.

**Author(s)**

Thomas Carroll and Rory Stark

**See Also**

[ChIPQC-package](#), [ChIPQCexperiment](#), [ChIPQCsample](#)

**Examples**

```
data(example_QCexperiment)
plotCoverageHist(exampleExp, facetBy=c("Tissue", "Factor"))
plotCoverageHist(QCsample(exampleExp, 1))
```

---

plotFribl-methods      *Generate fraction of reads in blacklist plot*

---

**Description**

Generate fraction of reads in blacklist plot.

**Methods**

signature(object = "ChIPQCexperiment", type="barstacked", facet=T, facetBy=c("Tissue", "Factor"), AsPercent=TRUE)  
Generate fraction of reads in blacklist plots for all the samples in an experiment.

signature(object = "list", type="barstacked", facet=T, facetBy=c("Sample"), AsPercent=TRUE)  
Generate fraction of reads in blacklist plots for all the samples in a list of ChIPQCsample objects..

signature(object = "ChIPQCsample", type="barstacked", AsPercent=TRUE) Generate fraction of reads in blacklist plots for a sample.

**Note**

plotFribl uses ggplot2 for plotting, and returns a ggplot2 plot dataframe.

**Author(s)**

Thomas Carroll and Rory Stark

**See Also**

[ChIPQC-package](#), [ChIPQCexperiment](#), [ChIPQCsample](#)

**Examples**

```
data(example_QCexperiment)
plotFribl(exampleExp)
plotFribl(QCsample(exampleExp,1))
```

---

plotFrip-methods      *Generate fraction of reads in peaks plot*

---

**Description**

Generate fraction of reads in peaks plot.

**Methods**

signature(object = "ChIPQCexperiment", type="barstacked", facet=T, facetBy=c("Tissue", "Factor"), AsPercent=TRUE)  
Generate fraction of reads in peaks plots for all the samples in an experiment.

signature(object = "list", type="barstacked", facet=T, facetBy=c("Sample"), AsPercent=TRUE)  
Generate fraction of reads in peaks plots for all the samples in a list of ChIPQCsample objects.

signature(object = "ChIPQCsample", type="barstacked", facet=T, facetBy=c("Tissue", "Factor"), AsPercent=TRUE)  
Generate fraction of reads in peaks plots for a sample.

**Note**

plotFrip uses ggplot2 for plotting, and returns a ggplot2 plot dataframe.

**Author(s)**

Thomas Carroll and Rory Stark

**See Also**

[ChIPQC-package](#), [ChIPQCexperiment](#), [ChIPQCsample](#)

**Examples**

```
data(example_QCexperiment)
plotFrip(exampleExp)
plotFrip(QCsample(exampleExp,1))
```

---

plotPeakProfile-methods

*Generate peak profile plots*

---

**Description**

Generate peak profile plots.

**Methods**

signature(object = "ChIPQCexperiment", method) Generate peak profile plots for all the samples in an experiment.

signature(object = "list", method) Generate peak profile plots for all the samples in a list of ChIPQCsample objects..

signature(object = "ChIPQCsample", method) Generate peak profile plots for a sample.

**Note**

plotPeakProfile uses ggplot2 for plotting, and returns a ggplot2 plot dataframe.

**Author(s)**

Thomas Carroll and Rory Stark

**See Also**

[ChIPQC-package](#), [ChIPQCexperiment](#), [ChIPQCsample](#)

**Examples**

```
data(example_QCexperiment)
plotCoverageHist(exampleExp, facetBy=c("Tissue", "Factor"))
plotCoverageHist(QCsample(exampleExp, 1))
data(tamoxifen_QC)
plotCoverageHist(tamoxifen, facetBy=c("Tissue", "Condition"))
```

---

plotPrincomp-methods *Generate Principal Components Analysis plot for ChIP samples*

---

**Description**

Generate principal components analysis plot for ChIP samples.

**Methods**

`signature(object = "ChIPQCexperiment", attributes, ...)` Generate principal components analysis plot, for all the samples in an experiment that are associated with a peakset.

`attributes` character string, or vector of character strings, representing metadata field names, for use grouping samples by color  
`...` additional parameters passed to [dba.plotPCA](#)

**Note**

`plotPrincomp` uses [dba.plotPCA](#) for plotting.

**Author(s)**

Rory Stark and Thomas Carroll

**See Also**

[ChIPQC-package](#), [ChIPQCexperiment](#), [dba.plotPCA](#)

**Examples**

```
data(tamoxifen_QC)
plotPrincomp(tamoxifen, attributes=c("Condition"))
```

---

plotRap-methods	<i>Generate reads in peaks plot</i>
-----------------	-------------------------------------

---

## Description

Generate reads in peaks plot.

## Methods

`signature(object = "ChIPQCexperiment", type="barstacked", facet=T, facetBy=c("Tissue", "Factor"))`  
Generate reads in peaks plots for all the samples in an experiment.

`signature(object = "list", type="barstacked", facet=T, facetBy=c("Sample"))` Generate reads in peaks plots for all the samples in a list of ChIPQCsample objects.

`signature(object = "ChIPQCexperiment", type="barstacked", facet=T, facetBy=c("Tissue", "Factor"))`  
Generate reads in peaks plots for all the samples in an experiment.

`signature(object = "ChIPQCsample", type="barstacked", facet=T, facetBy=c("Tissue", "Factor"))`  
Generate reads in peaks plots for a sample.

## Note

plotRap uses ggplot2 for plotting, and returns a ggplot2 plot dataframe.

## Author(s)

Thomas Carroll and Rory Stark

## See Also

[ChIPQC-package](#), [ChIPQCexperiment](#), [ChIPQCsample](#)

## Examples

```
data(example_QCexperiment)
plotRap(exampleExp, facetBy=c("Tissue", "Factor"))
plotRap(QCsample(exampleExp, 1))
```

---

plotRegi-methods      *Generate relative enrichment of genomic features plot*

---

**Description**

Generate relative enrichment of genomic features plot.

**Methods**

signature(object = "ChIPQCexperiment", facet=T, facetBy=c("Tissue", "Factor")) Generate relative enrichment of genomic features plots for all the samples in an experiment.

signature(object = "list", facet=T, facetBy=c("Sample")) Generate relative enrichment of genomic features plots for all the samples in a list of ChIPQCsample objects.

signature(object = "ChIPQCsample") Generate relative enrichment of genomic features plots for a sample.

**Note**

plotRegi uses ggplot2 for plotting, and returns a ggplot2 plot dataframe.

**Author(s)**

Thomas Carroll and Rory Stark

**See Also**

[ChIPQC-package](#), [ChIPQCexperiment](#), [ChIPQCsample](#)

**Examples**

```
data(example_QCexperiment)
plotRegi(exampleExp, facetBy=c("Tissue", "Factor"))
plotRegi(QCsample(exampleExp, 1))
data(tamoxifen_QC)
plotRegi(tamoxifen, facetBy=c("Tissue", "Condition"))
```

---

plotSSD-methods      *Generate SSD plot*

---

**Description**

Generate SSD metric plot. If blacklists supplied, will generate SSD prior and post blacklisting

**Methods**

`signature(object = "ChIPQCexperiment", facet=T, facetBy=c("Tissue", "Factor"))` Generate SSD metric plot for all samples in experiment. If blacklists supplied, will generate SSD prior and post blacklisting

`signature(object = "list", facet=T, facetBy=c("Tissue", "Factor"))` Generate SSD metric plot for list of samples in experiment. If blacklists supplied, will generate SSD prior and post blacklisting

`signature(object = "ChIPQCsample")` Generate SSD metric plot for single sample. If blacklists supplied, will generate SSD prior and post blacklisting

**Note**

`plotSSD` uses `ggplot2` for plotting, and returns a `ggplot2` plot `gg` object.

**Author(s)**

Thomas Carroll and Rory Stark

**See Also**

[ChIPQC-package](#), [ChIPQCexperiment](#), [ChIPQCsample](#)

**Examples**

```
data(example_QCexperiment)
plotSSD(exampleExp, facetBy=c("Tissue", "Factor"))
plotSSD(QCsample(exampleExp, 1))
plotSSD(QCsample(exampleExp)[1:3])
data(tamoxifen_QC)
plotSSD(tamoxifen, facetBy=c("Tissue", "Condition"))
```

---

QCAnnotation-methods *Retrieve an annotation description, or a processed annotation*

---

**Description**

Retrieve an annotation description, or a processed annotation, from a [ChIPQCexperiment](#) object.

**Methods**

`signature(object = "ChIPQCexperiment", bRetrieve=FALSE)` Retrieve the annotation. If `bRetrieve=FALSE` (default), the character string describing the annotation is returned (currently only "hg19" is supported). If `bRetrieve=TRUE`, a processed annotation is returned (in the form of a list). This can be used in subsequent calls to [ChIPQC](#) and/or [ChIPQCsample](#) for efficiency purposes.

**Author(s)**

Rory Stark and Thomas Carroll

**See Also**

[ChIPQC-package](#), [ChIPQCsample](#), [ChIPQCexperiment](#)

**Examples**

```
data(example_QCexperiment)
QCannotation(exampleExp)
```

---

QCcontrol-methods

*Retrieve control objects associated with a sample*

---

**Description**

Retrieve the [ChIPQCsample](#) objects representing controls, or a specific [ChIPQCsample](#) representing the control for a specific sample, from a [ChIPQCexperiment](#) object.

**Methods:**

`signature(object = "ChIPQCexperiment", sampleID)` Get the control sample (as a [ChIPQCsample](#) object) associated with a ChIP sample, or, if `sampleID` is missing, a list of all samples used as controls. `sampleID` is a character string or an integer.

**Author(s)**

Rory Stark and Thomas Carroll

**See Also**

[ChIPQC-package](#), [ChIPQCsample](#), [ChIPQCexperiment](#)

**Examples**

```
data(tamoxifen_QC)
controls = QCcontrol(tamoxifen)
length(controls)
names(controls)
controls[[1]]
bt474control = QCcontrol(tamoxifen,"BT4741")
bt474control
```



---

QCdba-methods

*Retrieve the [DBA-object](#) object associated with an experiment*

---

### Description

Retrieve the [DBA-object](#) object associated with a [ChIPQCexperiment](#) object.

### Methods:

`signature(object = "ChIPQCexperiment")` Retrieves the [DBA-object](#) object associated with a [ChIPQCexperiment](#). This object can be used with [DiffBind](#) functions to further analyse a ChIP-seq experiment, including performing a differential binding analysis.

### Author(s)

Rory Stark and Thomas Carroll

### See Also

[ChIPQC-package](#), [ChIPQCexperiment](#), [DiffBind](#), [dba](#)

### Examples

```
data(tamoxifen_QC)
tamoxifenDBA = QCdba(tamoxifen)
## library(DiffBind)
## tamoxifenDBA
```

---

QCmetadata-methods

*Retrieve metadata associated with an experiment*

---

### Description

Retrieve metadata for a [ChIPQCexperiment](#) object.

### Methods:

`signature(object = "ChIPQCexperiment")` Retrieve a data frame containing metadata for all the samples in a ChIP-seq experiment represented by a [ChIPQCexperiment](#) object.

`signature(object = "list")` Retrieve a data frame containing metadata for all the samples in a list of [ChIPQCsample](#) objects.

### Author(s)

Rory Stark and Thomas Carroll

**See Also**

[ChIPQC-package](#), [ChIPQCsample](#), [ChIPQCexperiment](#)

**Examples**

```
data(tamoxifen_QC)
meta = QCmetadata(tamoxifen)
meta
```

---

QCmetrics-methods	<i>Retrieve consolidated set of QC metrics</i>
-------------------	--

---

**Description**

Retrieves a consolidated set of QC metrics.

**Methods**

`signature(object = "ChIPQCexperiment")` Retrieves a matrix of QC metrics for all the samples in an experiment, with a column of values for each sample.

`signature(object = "list")` Retrieves a matrix of QC metrics for all the samples in a list of ChIPQCsample objects, with a column of values for each sample.

`signature(object = "ChIPQCsample")` Retrieves a vector of QC metrics for a sample.

**Author(s)**

Rory Stark and Thomas Carroll

**See Also**

[ChIPQC-package](#), [ChIPQCsample](#), [ChIPQCexperiment](#)

**Examples**

```
data(example_QCexperiment)
QCmetrics(exampleExp)

data(tamoxifen_QC)
QCmetrics(tamoxifen)

QCmetrics(QCsample(tamoxifen,1))
```

---

QCsample-methods      *Retrieve sample objects associated with an experiment*

---

**Description**

Retrieves a list of [ChIPQCsample](#) objects, or one specific [ChIPQCsample](#) object, from a [ChIPQCexperiment](#) object.

**Methods**

signature(object = "ChIPQCexperiment", sampleID) If sampleID is missing or equal to 0, the full list of [ChIPQCsample](#) objects is returned. If sampleID is an integer n, the [ChIPQCsample](#) object corresponding to the nth sample is returned. If sampleID is a character string, the [ChIPQCsample](#) object corresponding to the sample with that ID is returned.

**Author(s)**

Rory Stark and Thomas Carroll

**See Also**

[ChIPQC-package](#), [ChIPQCsample](#), [ChIPQCexperiment](#)

**Examples**

```
data(example_QCexperiment)
samples = QCsample(exampleExp)
length(samples)
names(samples)
samples$CTCF_1
```

---

readlength-methods      *Retrieve read length values*

---

**Description**

Retrieve read length values.

**Methods**

signature(object = "ChIPQCexperiment") Retrieve a vector of read length values, one for each sample in an experiment.

signature(object = "list") Retrieve a vector of read length values, one for each sample in a list of [ChIPQCsample](#) objects.

signature(object = "ChIPQCsample") Retrieve the read length value for a sample.

**Author(s)**

Thomas Carroll and Rory Stark

**See Also**

[ChIPQC-package](#), [ChIPQCexperiment](#), [ChIPQCsample](#)

**Examples**

```
data(example_QCexperiment)
readlength(exampleExp)
readlength(QCsample(exampleExp,1))
```

---

ReadLengthFragmentLengthCrossCoverage-methods

*Retrieve the cross coverage values when extending reads to the optimal fragment length.*

---

**Description**

Retrieve the cross coverage values when extending reads to the optimal fragment length.

**Methods**

signature(object = "ChIPQCexperiment") Retrieve a vector of cross-coverage values for all samples in a ChIP-seq experiment, when all reads are shifted by the optimal fragment length (the maximum cross-coverage value).

signature(object = "list") Retrieve a vector of cross-coverage values for all samples in a list of ChIPQCsample objects, when all reads are shifted by the optimal fragment length (the maximum cross-coverage value).

signature(object = "ChIPQCsample") Retrieve the cross-coverage value for a sample, when all reads are shifted by the optimal fragment length (the maximum cross-coverage value).

**Author(s)**

Thomas Carroll and Rory Stark

**See Also**

[ChIPQC-package](#), [ChIPQCexperiment](#), [ChIPQCsample](#)

**Examples**

```
data(example_QCexperiment)
FragmentLengthCrossCoverage(exampleExp)
FragmentLengthCrossCoverage(QCsample(exampleExp,1))
```

---

ReadLengthReadLengthCrossCoverage-methods

*Retrieve the cross coverage values without extending reads*

---

### **Description**

Retrieve the cross coverage values without extending reads.

### **Methods**

signature(object = "ChIPQCexperiment") Retrieve a vector of cross-coverage values for all samples in a ChIP-seq experiment, with no shift.

signature(object = "list") Retrieve a vector of cross-coverage values for all samples in a list of ChIPQCsample objects, with no shift.

signature(object = "ChIPQCsample") Retrieve the cross-coverage value for a sample, with no shift.

### **Author(s)**

Thomas Carroll and Rory Stark

### **See Also**

[ChIPQC-package](#), [ChIPQCexperiment](#), [ChIPQCsample](#)

### **Examples**

```
data(example_QCexperiment)
ReadLengthCrossCoverage(exampleExp)
ReadLengthCrossCoverage(QCsample(exampleExp,1))
```

---

reads-methods

*Retrieve numbers of reads*

---

### **Description**

Retrieve the numbers of reads.

**Methods**

`signature(object = "ChIPQCexperiment", bFiltered)` Retrieve a vector of the numbers of reads for each sample in an experiment. If `bFiltered=TRUE` (or is missing), this will be the number of reads that pass the mapping quality filter for each sample. if `bFiltered=FALSE`, it will be the total number of reads for each sample.

`signature(object = "list", bFiltered)` Retrieve a vector of the numbers of reads for each sample in a list of `ChIPQCsample` objects. If `bFiltered=TRUE` (or is missing), this will be the number of reads that pass the mapping quality filter for each sample. if `bFiltered=FALSE`, it will be the total number of reads for each sample.

`signature(object = "ChIPQCsample", bFiltered)` Retrieve the number of reads for a sample. If `bFiltered=TRUE` (or is missing), this will be the number of reads that pass the mapping quality filter. if `bFiltered=FALSE`, it will be the total number of reads for the sample.

**Author(s)**

Thomas Carroll and Rory Stark

**See Also**

[ChIPQC-package](#), [ChIPQCexperiment](#), [ChIPQCsample](#)

**Examples**

```
data(example_QCexperiment)
reads(exampleExp)
reads(QCsample(exampleExp,1))
```

---

regi-methods

*Retrieve genomic profile information*

---

**Description**

Retrieve genomic profile information in terms of relative enrichment over background genomic distribution.

**Methods**

`signature(object = "ChIPQCexperiment")` Retrieve a matrix of relative enrichment values for a variety of genomic features. Each column represents the enrichment values for one sample in the experiment.

`signature(object = "list")` Retrieve a matrix of relative enrichment values for a variety of genomic features. Each column represents the enrichment values for one sample in a list of `ChIPQCsample` objects.

`signature(object = "ChIPQCsample")` Retrieve a vector of relative enrichment values for a variety of genomic features for a sample. Relative enrichment is computed as the proportion of reads overlapping a genomic feature type compared to the overall proportion of base pairs in the genome comprising those features. Genomic features include:

3UTRs	3' UTRs
5UTRs	5' UTRs
Introns	Intronic (non-coding) portions of gene bodies
Transcripts	Transcribed regions, including exons
Promoters500	500bp regions immediately upstream of annotated TSSs
Promoters2000to500	2500bp regions from 2000bp immediately upstream of annotated TSSs to 500bp downstream
Promoters20000to2000	22000bp regions from 20000bp immediately upstream of annotated TSSs to 2000bp downstream

**Author(s)**

Thomas Carroll and Rory Stark

**See Also**

[ChIPQC-package](#), [ChIPQCexperiment](#), [ChIPQCsample](#)

**Examples**

```
data(example_QCexperiment)
genomicprofile = regi(exampleExp)
heatmap(genomicprofile)
regi(QCsample(exampleExp,1))
```

---

RelativeCrossCoverage-methods

*Retrieve the relative cross coverage values for a range of shift sizes*

---

**Description**

Retrieve the relative cross-coverage values for a range of shift sizes

**Methods**

`signature(object = "ChIPQCexperiment")` Retrieve a vector of relative cross-coverage values for all samples in a ChIP-seq experiment, computed based on the maximal value (when extending the reads to the optimal fragment length) versus the cross-coverage values using non-extended reads.

`signature(object = "list")` Retrieve a vector of relative cross-coverage values for all samples in a list of ChIPQCsample objects, computed based on the maximal value (when extending the reads to the optimal fragment length) versus the cross-coverage values using non-extended reads.

`signature(object = "ChIPQCsample")` Retrieve the relative cross-coverage value for a sample, computed based on the maximal value (when extending the reads to the optimal fragment length) versus the cross-coverage values using non-extended reads.

**Author(s)**

Thomas Carroll and Rory Stark

**See Also**

[ChIPQC-package](#), [ChIPQCexperiment](#), [ChIPQCsample](#)

**Examples**

```
data(example_QCexperiment)
RelativeCrossCoverage(exampleExp)
RelativeCrossCoverage(QCsample(exampleExp,1))
```

---

ribl-methods

*Retrieve numbers of reads overlapping blacklisted regions*

---

**Description**

Retrieve the numbers of reads overlapping blacklisted regions.

**Methods**

signature(object = "ChIPQCexperiment") Retrieve a vector of the numbers of reads overlapping blacklisted regions for each sample in an experiment.

signature(object = "list") Retrieve a vector of the numbers of reads overlapping blacklisted regions for each sample in a list of ChIPQCsample objects.

signature(object = "ChIPQCsample") Retrieve the number of reads overlapping blacklisted regions in a sample.

**Author(s)**

Thomas Carroll and Rory Stark

**See Also**

[ChIPQC-package](#), [ChIPQCexperiment](#), [ChIPQCsample](#)

**Examples**

```
data(example_QCexperiment)
ribl(exampleExp)
ribl(QCsample(exampleExp,1))
```



---

rip-methods	<i>Retrieve numbers of reads overlapping peaks</i>
-------------	--

---

**Description**

Retrieve the numbers of reads overlapping peaks.

**Methods**

`signature(object = "ChIPQCexperiment")` Retrieve a vector of the numbers of reads overlapping peaks for each sample in an experiment.

`signature(object = "list")` Retrieve a vector of the numbers of reads overlapping peaks for each sample in a list of `ChIPQCsample` objects.

`signature(object = "ChIPQCsample")` Retrieve the number of reads overlapping peaks in a sample.

**Author(s)**

Thomas Carroll and Rory Stark

**See Also**

[ChIPQC-package](#), [ChIPQCexperiment](#), [ChIPQCsample](#)

**Examples**

```
data(example_QCexperiment)
rip(exampleExp)
rip(QCsample(exampleExp,1))
```

---

ssd-methods	<i>Retrieve SSD (squared sum of deviations) values of peak coverage</i>
-------------	---

---

**Description**

Retrieve SSD (squared sum of deviations) values of peak coverage density.

**Methods**

`signature(object = "ChIPQCexperiment")` Retrieve a vector of SSD values, one for each sample in an experiment.

`signature(object = "list")` Retrieve a vector of SSD values, one for each sample in a list of `ChIPQCsample` objects.

`signature(object = "ChIPQCsample")` Retrieve the SSD (squared sum of deviations) for a sample, computed from the standard deviation of the [coveragehistogram](#).

**Note**

uses the SSD calculation from the chipseq package.

**Author(s)**

Thomas Carroll and Rory Stark

**See Also**

[ChIPQC-package](#), [ChIPQCexperiment](#), [ChIPQCsample](#)

**Examples**

```
data(example_QCexperiment)
ssd(exampleExp)
ssd(QCsample(exampleExp,1))
```

# Index

- \* **classes**
  - ChIPQCexperiment-class, 9
  - ChIPQCsample-class, 12
- \* **datasets**
  - ChIPQC-data, 8
- \* **methods**
  - averagepeaksignal-methods, 4
  - ChIPQCreport-methods, 11
  - coveragehistogram-methods, 15
  - crosscoverage-methods, 16
  - duplicateRate-methods, 17
  - duplicates-methods, 18
  - flagtagcounts-methods, 19
  - fragmentlength-methods, 19
  - frip-methods, 20
  - mapped-methods, 21
  - Normalisedaveragepeaksignal-methods, 22
  - peaks-methods, 23
  - plotCC-methods, 23
  - plotCorHeatmap-methods, 24
  - plotCoverageHist-methods, 25
  - plotFribl-methods, 26
  - plotFrip-methods, 26
  - plotPeakProfile-methods, 27
  - plotPrincomp-methods, 28
  - plotRap-methods, 29
  - plotRegi-methods, 30
  - plotSSD-methods, 30
  - QCannotation-methods, 31
  - QCcontrol-methods, 32
  - QCdba-methods, 33
  - QCmetadata-methods, 33
  - QCmetrics-methods, 34
  - QCsample-methods, 35
  - readlength-methods, 35
  - ReadLengthFragmentLengthCrossCoverage-methods, 36
  - ReadLengthReadLengthCrossCoverage-methods, 37
  - reads-methods, 37
  - regi-methods, 38
  - RelativeCrossCoverage-methods, 39
  - ribl-methods, 40
  - rip-methods, 41
  - ssd-methods, 41
- averagepeaksignal, 9, 14
- averagepeaksignal
  - (averagepeaksignal-methods), 4
- averagepeaksignal, ChIPQCexperiment-method
  - (averagepeaksignal-methods), 4
- averagepeaksignal, ChIPQCsample-method
  - (averagepeaksignal-methods), 4
- averagepeaksignal, list-method
  - (averagepeaksignal-methods), 4
- averagepeaksignal-methods, 4
- BiocParallel, 7
- blacklist\_hg19, 6, 13
- blacklist\_hg19 (ChIPQC-data), 8
- bplapply, 7
- ChIPQC, 3, 5, 8, 9, 31
- ChIPQC-data, 8
- ChIPQC-package, 3, 4, 7, 10, 12, 15–42
- ChIPQCexperiment, 3–8, 11, 12, 15–42
- ChIPQCexperiment
  - (ChIPQCexperiment-class), 9
- ChIPQCexperiment-class, 9
- ChIPQCreport, 10
- ChIPQCreport (ChIPQCreport-methods), 11
- ChIPQCreport, ChIPQCexperiment-method
  - (ChIPQCreport-methods), 11
- ChIPQCreport, ChIPQCsample-method
  - (ChIPQCreport-methods), 11
- ChIPQCreport, list-method
  - (ChIPQCreport-methods), 11
- ChIPQCreport-methods, 11

- ChIPQCsamples, [3](#), [4](#), [6](#), [7](#), [9–12](#), [15–32](#), [34–42](#)
- ChIPQCsamples (ChIPQCsamples-class), [12](#)
- ChIPQCsamples-class, [12](#)
- coveragehistogram, [9](#), [14](#), [41](#)
- coveragehistogram
  - (coveragehistogram-methods), [15](#)
- coveragehistogram,ChIPQCExperiment-method
  - (coveragehistogram-methods), [15](#)
- coveragehistogram,ChIPQCsamples-method
  - (coveragehistogram-methods), [15](#)
- coveragehistogram,list-method
  - (coveragehistogram-methods), [15](#)
- coveragehistogram-methods, [15](#)
- crosscoverage, [9](#), [14](#)
- crosscoverage (crosscoverage-methods), [16](#)
- crosscoverage,ChIPQCExperiment-method
  - (crosscoverage-methods), [16](#)
- crosscoverage,ChIPQCsamples-method
  - (crosscoverage-methods), [16](#)
- crosscoverage,list-method
  - (crosscoverage-methods), [16](#)
- crosscoverage-methods, [16](#)
  
- dba, [6](#), [33](#)
- DBA-object, [33](#)
- dba.count, [6](#), [7](#)
- dba.peakset, [5](#)
- dba.plotHeatmap, [24](#), [25](#)
- dba.plotPCA, [28](#)
- DiffBind, [3](#), [5](#), [7–10](#), [33](#)
- duplicateRate, [10](#), [14](#)
- duplicateRate (duplicateRate-methods), [17](#)
- duplicateRate,ChIPQCExperiment-method
  - (duplicateRate-methods), [17](#)
- duplicateRate,ChIPQCsamples-method
  - (duplicateRate-methods), [17](#)
- duplicateRate,list-method
  - (duplicateRate-methods), [17](#)
- duplicateRate-methods, [17](#)
- duplicates, [10](#), [14](#)
- duplicates (duplicates-methods), [18](#)
- duplicates,ChIPQCExperiment-method
  - (duplicates-methods), [18](#)
- duplicates,ChIPQCsamples-method
  - (duplicates-methods), [18](#)
- duplicates,list-method
  - (duplicates-methods), [18](#)
  
- duplicates-methods, [18](#)
- example\_QCexperiment (ChIPQC-data), [8](#)
- exampleExp (ChIPQC-data), [8](#)
  
- flagtagcounts, [9](#), [14](#)
- flagtagcounts (flagtagcounts-methods), [19](#)
- flagtagcounts,ChIPQCExperiment-method
  - (flagtagcounts-methods), [19](#)
- flagtagcounts,ChIPQCsamples-method
  - (flagtagcounts-methods), [19](#)
- flagtagcounts,list-method
  - (flagtagcounts-methods), [19](#)
- flagtagcounts-methods, [19](#)
- fragmentlength, [9](#), [14](#)
- fragmentlength
  - (fragmentlength-methods), [19](#)
- fragmentlength,ChIPQCExperiment-method
  - (fragmentlength-methods), [19](#)
- fragmentlength,ChIPQCsamples-method
  - (fragmentlength-methods), [19](#)
- fragmentlength,list-method
  - (fragmentlength-methods), [19](#)
- fragmentlength-methods, [19](#)
- FragmentLengthCrossCoverage, [9](#), [14](#)
- FragmentLengthCrossCoverage
  - (ReadLengthFragmentLengthCrossCoverage-methods), [36](#)
- FragmentLengthCrossCoverage,ChIPQCExperiment-method
  - (ReadLengthFragmentLengthCrossCoverage-methods), [36](#)
- FragmentLengthCrossCoverage,ChIPQCsamples-method
  - (ReadLengthFragmentLengthCrossCoverage-methods), [36](#)
- FragmentLengthCrossCoverage,list-method
  - (ReadLengthFragmentLengthCrossCoverage-methods), [36](#)
- FragmentLengthCrossCoverage-methods
  - (ReadLengthFragmentLengthCrossCoverage-methods), [36](#)
  
- frip, [9](#), [14](#)
- frip (frip-methods), [20](#)
- frip,ChIPQCExperiment-method
  - (frip-methods), [20](#)
- frip,ChIPQCsamples-method
  - (frip-methods), [20](#)
- frip,list-method (frip-methods), [20](#)
- frip-methods, [20](#)

- GRanges, [6](#), [8](#), [12–14](#), [23](#)
- GRangesList, [23](#)
- list, [9](#), [11](#)
- mapped, [9](#), [14](#)
- mapped (mapped-methods), [21](#)
- mapped, ChIPQCexperiment-method  
(mapped-methods), [21](#)
- mapped, ChIPQCsample-method  
(mapped-methods), [21](#)
- mapped, list-method (mapped-methods), [21](#)
- mapped-methods, [21](#)
- Normalisedaveragepeaksignal, [10](#), [14](#)
- Normalisedaveragepeaksignal  
(Normalisedaveragepeaksignal-methods),  
[22](#)
- Normalisedaveragepeaksignal, ChIPQCexperiment-method  
(Normalisedaveragepeaksignal-methods),  
[22](#)
- Normalisedaveragepeaksignal, ChIPQCsample-method  
(Normalisedaveragepeaksignal-methods),  
[22](#)
- Normalisedaveragepeaksignal, list-method  
(Normalisedaveragepeaksignal-methods),  
[22](#)
- Normalisedaveragepeaksignal-methods,  
[22](#)
- peaks, [10](#), [14](#)
- peaks (peaks-methods), [23](#)
- peaks, ChIPQCexperiment-method  
(peaks-methods), [23](#)
- peaks, ChIPQCsample-method  
(peaks-methods), [23](#)
- peaks, list-method (peaks-methods), [23](#)
- peaks-methods, [23](#)
- plotCC, [10](#), [14](#)
- plotCC (plotCC-methods), [23](#)
- plotCC, ChIPQCexperiment-method  
(plotCC-methods), [23](#)
- plotCC, ChIPQCsample-method  
(plotCC-methods), [23](#)
- plotCC, list-method (plotCC-methods), [23](#)
- plotCC-methods, [23](#)
- plotCorHeatmap, [10](#)
- plotCorHeatmap  
(plotCorHeatmap-methods), [24](#)
- plotCorHeatmap, ChIPQCexperiment-method  
(plotCorHeatmap-methods), [24](#)
- plotCorHeatmap-methods, [24](#)
- plotCoverageHist, [10](#), [14](#)
- plotCoverageHist  
(plotCoverageHist-methods), [25](#)
- plotCoverageHist, ChIPQCexperiment-method  
(plotCoverageHist-methods), [25](#)
- plotCoverageHist, ChIPQCsample-method  
(plotCoverageHist-methods), [25](#)
- plotCoverageHist, list-method  
(plotCoverageHist-methods), [25](#)
- plotCoverageHist-methods, [25](#)
- plotFribl, [10](#), [14](#)
- plotFribl (plotFribl-methods), [26](#)
- plotFribl, ChIPQCexperiment-method  
(plotFribl-methods), [26](#)
- plotFribl, ChIPQCsample-method  
(plotFribl-methods), [26](#)
- plotFribl, list-method  
(plotFribl-methods), [26](#)
- plotFribl-methods, [26](#)
- plotFrip (plotFrip-methods), [26](#)
- plotFrip, ChIPQCexperiment-method  
(plotFrip-methods), [26](#)
- plotFrip, ChIPQCsample-method  
(plotFrip-methods), [26](#)
- plotFrip, list-method  
(plotFrip-methods), [26](#)
- plotFrip-methods, [26](#)
- plotPeakProfile, [10](#), [14](#)
- plotPeakProfile  
(plotPeakProfile-methods), [27](#)
- plotPeakProfile, ChIPQCexperiment-method  
(plotPeakProfile-methods), [27](#)
- plotPeakProfile, ChIPQCsample-method  
(plotPeakProfile-methods), [27](#)
- plotPeakProfile, list-method  
(plotPeakProfile-methods), [27](#)
- plotPeakProfile-methods, [27](#)
- plotPrincomp, [10](#)
- plotPrincomp (plotPrincomp-methods), [28](#)
- plotPrincomp, ChIPQCexperiment-method  
(plotPrincomp-methods), [28](#)
- plotPrincomp-methods, [28](#)
- plotRap, [10](#), [14](#)
- plotRap (plotRap-methods), [29](#)
- plotRap, ChIPQCexperiment-method

- (plotRap-methods), 29
- plotRap,ChIPQCsample-method  
(plotRap-methods), 29
- plotRap,list-method (plotRap-methods),  
29
- plotRap-methods, 29
- plotRegi, 10, 14
- plotRegi (plotRegi-methods), 30
- plotRegi,ChIPQCexperiment-method  
(plotRegi-methods), 30
- plotRegi,ChIPQCsample-method  
(plotRegi-methods), 30
- plotRegi,list-method  
(plotRegi-methods), 30
- plotRegi-methods, 30
- plotSSD (plotSSD-methods), 30
- plotSSD,ChIPQCexperiment-method  
(plotSSD-methods), 30
- plotSSD,ChIPQCsample-method  
(plotSSD-methods), 30
- plotSSD,list-method (plotSSD-methods),  
30
- plotSSD-methods, 30
  
- QCAnnotation, 6, 9, 12
- QCAnnotation (QCAnnotation-methods), 31
- QCAnnotation,ChIPQCexperiment-method  
(QCAnnotation-methods), 31
- QCAnnotation-methods, 31
- QCcontrol, 9
- QCcontrol (QCcontrol-methods), 32
- QCcontrol,ChIPQCexperiment-method  
(QCcontrol-methods), 32
- QCcontrol-methods, 32
- QCdba, 9
- QCdba (QCdba-methods), 33
- QCdba,ChIPQCexperiment-method  
(QCdba-methods), 33
- QCdba-methods, 33
- QCmetadata, 9
- QCmetadata (QCmetadata-methods), 33
- QCmetadata,ChIPQCexperiment-method  
(QCmetadata-methods), 33
- QCmetadata,list-method  
(QCmetadata-methods), 33
- QCmetadata-methods, 33
- QCmetrics, 9
- QCmetrics (QCmetrics-methods), 34
  
- QCmetrics,ChIPQCexperiment-method  
(QCmetrics-methods), 34
- QCmetrics,ChIPQCsample-method  
(QCmetrics-methods), 34
- QCmetrics,list-method  
(QCmetrics-methods), 34
- QCmetrics-methods, 34
- QCsample, 9
- QCsample (QCsample-methods), 35
- QCsample,ChIPQCexperiment-method  
(QCsample-methods), 35
- QCsample-methods, 35
  
- readlength, 10, 14
- readlength (readlength-methods), 35
- readlength,ChIPQCexperiment-method  
(readlength-methods), 35
- readlength,ChIPQCsample-method  
(readlength-methods), 35
- readlength,list-method  
(readlength-methods), 35
- readlength-methods, 35
- ReadLengthCrossCoverage, 10, 14
- ReadLengthCrossCoverage  
(ReadLengthReadLengthCrossCoverage-methods),  
37
- ReadLengthCrossCoverage,ChIPQCexperiment-method  
(ReadLengthReadLengthCrossCoverage-methods),  
37
- ReadLengthCrossCoverage,ChIPQCsample-method  
(ReadLengthReadLengthCrossCoverage-methods),  
37
- ReadLengthCrossCoverage,list-method  
(ReadLengthReadLengthCrossCoverage-methods),  
37
- ReadLengthCrossCoverage-methods  
(ReadLengthReadLengthCrossCoverage-methods),  
37
- ReadLengthFragmentLengthCrossCoverage-methods,  
36
- ReadLengthReadLengthCrossCoverage-methods,  
37
- reads, 10, 14
- reads (reads-methods), 37
- reads,ChIPQCexperiment-method  
(reads-methods), 37
- reads,ChIPQCsample-method  
(reads-methods), 37
- reads,list-method (reads-methods), 37

reads-methods, 37  
regi, 10, 14  
regi (regi-methods), 38  
regi,ChIPQCexperiment-method  
    (regi-methods), 38  
regi,ChIPQCsample-method  
    (regi-methods), 38  
regi,list-method (regi-methods), 38  
regi-methods, 38  
register, 7  
RelativeCrossCoverage, 10, 14  
RelativeCrossCoverage  
    (RelativeCrossCoverage-methods),  
    39  
RelativeCrossCoverage,ChIPQCexperiment-method  
    (RelativeCrossCoverage-methods),  
    39  
RelativeCrossCoverage,ChIPQCsample-method  
    (RelativeCrossCoverage-methods),  
    39  
RelativeCrossCoverage,list-method  
    (RelativeCrossCoverage-methods),  
    39  
RelativeCrossCoverage-methods, 39  
ribl, 10, 14  
ribl (ribl-methods), 40  
ribl,ChIPQCexperiment-method  
    (ribl-methods), 40  
ribl,ChIPQCsample-method  
    (ribl-methods), 40  
ribl,list-method (ribl-methods), 40  
ribl-methods, 40  
rip, 10, 14  
rip (rip-methods), 41  
rip,ChIPQCexperiment-method  
    (rip-methods), 41  
rip,ChIPQCsample-method (rip-methods),  
    41  
rip,list-method (rip-methods), 41  
rip-methods, 41  
  
show, 10, 14  
show,ChIPQCexperiment-method  
    (ChIPQCexperiment-class), 9  
show,ChIPQCsample-method  
    (ChIPQCsample-class), 12  
ssd, 10, 14  
ssd (ssd-methods), 41  
ssd,ChIPQCexperiment-method  
    (ssd-methods), 41  
ssd,ChIPQCsample-method (ssd-methods),  
    41  
ssd,list-method (ssd-methods), 41  
ssd-methods, 41  
tamoxifen, 8  
tamoxifen (ChIPQC-data), 8  
tamoxifen\_QC (ChIPQC-data), 8