

# Package ‘soGGi’

March 30, 2021

**Type** Package

**Title** Visualise ChIP-seq, MNase-seq and motif occurrence as aggregate plots Summarised Over Grouped Genomic Intervals

**Version** 1.22.0

**Date** 2019-27-03

**Author** Gopuraja Dharmalingam, Doug Barrows, Tom Carroll

**Maintainer** Tom Carroll <tc.infomatics@gmail.com>

**Description** The soGGi package provides a toolset to create genomic interval aggregate/summary plots of signal or motif occurrence from BAM and bigWig files as well as PWM, rlelist, GRanges and GAlignments Bioconductor objects. soGGi allows for normalisation, transformation and arithmetic operation on and between summary plot objects as well as grouping and subsetting of plots by GRanges objects and user supplied metadata. Plots are created using the GGplot2 library to allow user defined manipulation of the returned plot object. Coupled together, soGGi features a broad set of methods to visualise genomics data in the context of groups of genomic intervals such as genes, superenhancers and transcription factor binding events.

**biocViews** Sequencing, ChIPSeq, Coverage

**License** GPL (>= 3)

**LazyLoad** yes

**Depends** R (>= 3.2.0), BiocGenerics, SummarizedExperiment

**Imports** methods, reshape2, ggplot2, S4Vectors, IRanges, GenomeInfoDb, GenomicRanges, Biostrings, Rsamtools, GenomicAlignments, rtracklayer, preprocessCore, chipseq, BiocParallel

**Collate** 'allClasses.r' 'motifTools.R' 'peakTransforms.r' 'plots.R' 'soggi.R'

**VignetteBuilder** knitr

**Suggests** testthat, BiocStyle, knitr

**NeedsCompilation** no

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

**git\_branch** RELEASE\_3\_12

**git\_last\_commit** 7f949ee

**git\_last\_commit\_date** 2020-10-27

**Date/Publication** 2021-03-29

## R topics documented:

c,ChIPprofile-method	2
chipExampleBig	3
ChIPprofile-class	4
findconsensusRegions	5
groupByOverlaps	6
ik_Example	7
ik_Profiles	7
normalise	8
normaliseQuantiles	8
Ops,ChIPprofile,ChIPprofile-method	9
orientBy	10
plotRegion	11
pwmCov	12
pwmToCoverage	12
singleGRange	13
<b>Index</b>	<b>14</b>

---

c,ChIPprofile-method *Join, subset and manipulate ChIPprofile objects*

---

### Description

Join, subset and manipulate ChIPprofile objects

### Usage

```
## S4 method for signature 'ChIPprofile'
c(x, ..., recursive = FALSE)

## S4 method for signature 'ChIPprofile'
rbind(x, ..., deparse.level = 1)

## S4 method for signature 'ChIPprofile'
cbind(x, ..., deparse.level = 1)

## S4 method for signature 'ChIPprofile,ANY,missing'
x[[i, j, ...]]

## S4 method for signature 'ChIPprofile'
x$name
```

### Arguments

j	Should be missing
...	objects to be concatenated.
recursive	logical. If recursive = TRUE, the function recursively descends through lists (and pairlists) combining all their elements into a vector.
deparse.level	See ?base::cbind for a description of this argument.

x	object from which to extract element(s) or in which to replace element(s).
i	indices specifying elements to extract or replace. Indices are numeric or character vectors or empty (missing) or NULL. Numeric values are coerced to integer as by <code>as.integer</code> (and hence truncated towards zero). Character vectors will be matched to the <code>names</code> of the object (or for matrices/arrays, the <code>dimnames</code> ): see ‘Character indices’ below for further details.  For <code>[]</code> -indexing only: <code>i, j, ...</code> can be logical vectors, indicating elements/slices to select. Such vectors are recycled if necessary to match the corresponding extent. <code>i, j, ...</code> can also be negative integers, indicating elements/slices to leave out of the selection.  When indexing arrays by <code>[]</code> a single argument <code>i</code> can be a matrix with as many columns as there are dimensions of <code>x</code> ; the result is then a vector with elements corresponding to the sets of indices in each row of <code>i</code> .  An index value of NULL is treated as if it were <code>integer(0)</code> .
name	A literal character string or a <code>name</code> (possibly <code>backtick</code> quoted). For extraction, this is normally (see under ‘Environments’) partially matched to the <code>names</code> of the object.

**Value**

A ChIPprofile object

**Examples**

```
data(chipExampleBig)
x <- c(chipExampleBig[[1]],chipExampleBig[[2]])
y <- rbind(chipExampleBig[[1]],chipExampleBig[[2]])
```

---

chipExampleBig	<i>Example ChIPprofiles</i>
----------------	-----------------------------

---

**Description**

This dataset contains peaks from ChIP-signal over genes

**Usage**

```
data(chipExampleBig)
```

**Details**

- ChIPprofiles

**Value**

A ChIPprofile object

---

ChIPprofile-class      *The soggi function and ChIPprofile object.*

---

## Description

Manual for soggi and ChIPprofile object

The soggi function is the constructor for ChIPprofile objects.

## Usage

```
regionPlot(bamFile, testRanges, samplename = NULL, nOfWindows = 100,
  FragmentLength = 150, style = "point", distanceAround = NULL,
  distanceUp = NULL, distanceDown = NULL, distanceInRegionStart = NULL,
  distanceOutRegionStart = NULL, distanceInRegionEnd = NULL,
  distanceOutRegionEnd = NULL, paired = FALSE, normalize = "RPM",
  plotBy = "coverage", removeDup = FALSE, verbose = TRUE,
  format = "bam", seqlengths = NULL, forceFragment = NULL,
  method = "bin", genome = NULL, cutoff = 80, downSample = NULL,
  minFragmentLength = NULL, maxFragmentLength = NULL)
```

## Arguments

bamFile	Character vector for location of BAM file or bigWig, an rleList or PWM matrix.
testRanges	GRanges object or character vector of BED file location of regions to plot.
samplename	Character vector of sample name. Default is NULL.
nOfWindows	Number of windows to bin regions into for coverage calculations (Default 100)
FragmentLength	Integer vector Predicted or expected fragment length.
style	"Point" for per base pair plot, "percentOfRegion" for normalised length and "region" for combined plot
distanceAround	Distance around centre of region to be used for plotting
distanceUp	Distance upstream from centre of region to be used for plotting
distanceDown	Distance downstream from centre of region to be used for plotting
distanceInRegionStart	Distance into region start (5' for Watson/positive strand or notspecified strand Regions,3' for Crick/negatie strand regions) for plotting.
distanceOutRegionStart	Distance out from region start (5' for Watson/positive strand or notspecified strand Regions,3' for Crick/negatie strand regions) for plotting.
distanceInRegionEnd	Distance into region end (3' for Watson/positive strand or notspecified strand Regions,5' for Crick/negatie strand regions) for plotting.
distanceOutRegionEnd	Distance out from region end (3' for Watson/positive strand or notspecified strand Regions,5' for Crick/negatie strand regions) for plotting.
paired	Is data paired end
normalize	Calculate coverage as RPM. Presently only RPM available.

plotBy	Score to be used for plotting. Presently only coverage.
removeDup	Remove duplicates before calculating coverage.
verbose	TRUE or FALSE
format	character vector of "BAM", "BigWig", "RleList" or "PWM"
seqlengths	Chromosomes to be used. If missing will report all.
forceFragment	Centre fragment and force consistent fragment width.
method	Character vector of value "bp", "bin" or "spline". The bin method divides a region of interest into equal sized bins of number specified in nOfWindows. Coverage or counts are then summarised within these windows. The spline method creates a spline with the number of spline points as specified in nOfWindows argument.
downSample	Down sample BAM reads to this proportion of original.
genome	BSGenome object to be used when using PWM input.
cutoff	Cut-off for identifying motifs when using PWM input.
minFragmentLength	Remove fragments smaller than this.
maxFragmentLength	Remove fragments larger than this.

**Value**

ChIPprofile A ChIPprofile object.

**References**

See <http://bioinformatics.csc.mrc.ac.uk> for more details on soGGi workflows

**Examples**

```
data(chipExampleBig)
chipExampleBig
```

---

findconsensusRegions *Plot coverage of points or regions.*

---

**Description**

Plot coverage of points or regions.

Returns summits and summit scores after optional fragment length prediction and read extension

**Usage**

```
findconsensusRegions(testRanges, bamFiles = NULL, method = "majority",
  summit = "mean", resizepeak = "asw", overlap = "any",
  fragmentLength = NULL, NonPrimaryPeaks = list(withinsample = "drop",
  betweensample = "mean"))
```

```
summitPipeline(reads, peakfile, fragmentLength, readlength)
```

**Arguments**

testRanges	Named character vector of region locations
bamFiles	Named character vector of bamFile locations
method	Method to select reproducible summits to merge.
summit	Only mean available
resizepeak	Only asw available
overlap	Type of overlap to consider for finding consensus sites
fragmentLength	Predicted fragment length. Set to NULL to auto-calculate
NonPrimaryPeaks	A list of parameters to deal with non primary peaks in consensus regions.
peakfile	GRanges of genomic intervals to summit.
reads	Character vector of bamFile location or GAlignments object
readlength	Read length of alignments.

**Value**

Consensus A GRanges object of consensus regions with consensus summits.

Summits A GRanges object of summits and summit scores.

---

groupByOverlaps	<i>Create GRangeslist from all combinations of GRanges</i>
-----------------	--

---

**Description**

Create GRangeslist from all combinations of GRanges

**Usage**

```
groupByOverlaps(testRanges)
```

**Arguments**

testRanges	A named list of GRanges or a named GRangesList
------------	--

**Value**

groupedGRanges A named GRangesList object.

**Examples**

```
data(ik_Example)
gts <- groupByOverlaps(ik_Example)
```

---

`ik_Example`*Example Ikaros peaksets*

---

**Description**

This dataset contains peaks from Ikaros ChIP by two antibodies

**Usage**

```
data(ik_Example)
```

**Details**

- Ikpeaksets

**Value**

A list containing two GRanges objects

---

`ik_Profiles`*Example Ikaros signal over peaksets*

---

**Description**

This dataset contains signal over peaks from Ikaros ChIP by two antibodies

**Usage**

```
data(ik_Profiles)
```

**Details**

- ik\_Profiles

**Value**

A ChIPprofile object

---

 normalise

*Normalise ChIPprofiles*


---

### Description

Various normalisation methods for ChIPprofile objects

### Usage

```
## S4 method for signature 'ChIPprofile'
normalise(object)

## S4 method for signature 'ChIPprofile,character,numeric'
normalise(object = "ChIPprofile",
  method = "rpm", normFactors = NULL)
```

### Arguments

object	A ChIPprofile object
method	A character vector specifying normalisation method. Currently "rpm" for normalising signal for BAM to total reads, "quantile" to quantile normalise across samples, "signalInRegion" to normalise to proportion of signal within intervals, "normaliseSample" to normalise across samples and "normaliseRegions" to apply a normalisation across intervals.
normFactors	A numeric vector used to scale columns or rows.

### Value

A ChIPprofile object

### Author(s)

Thomas Carroll

### Examples

```
data(chipExampleBig)
normalise(chipExampleBig,method="quantile",normFactors=1)
```

---

 normaliseQuantiles

*Normalise quantile*


---

### Description

Quantile normalisation across bins/regions.



**Usage**

```
## S4 method for signature 'ChIPprofile'  
normaliseQuantiles(object)  
  
## S4 method for signature 'ChIPprofile'  
normaliseQuantiles(object = "ChIPprofile")
```

**Arguments**

object            A ChIPprofile object

**Value**

A ChIPprofile object containing normalised data

**Author(s)**

Thomas Carroll

**Examples**

```
data(chipExampleBig)  
normaliseQuantiles(chipExampleBig)
```

---

Ops,ChIPprofile,ChIPprofile-method  
*Arithmetic operations*

---

**Description**

Arithmetic operations

**Usage**

```
## S4 method for signature 'ChIPprofile,ChIPprofile'  
Ops(e1, e2)  
  
## S4 method for signature 'ChIPprofile,numeric'  
Ops(e1, e2)  
  
## S4 method for signature 'numeric,ChIPprofile'  
Ops(e1, e2)  
  
## S4 method for signature 'ChIPprofile'  
mean(x, ...)  
  
## S4 method for signature 'ChIPprofile'  
log2(x)  
  
## S4 method for signature 'ChIPprofile'  
log(x, base = exp(1))
```

**Arguments**

e1	ChIPprofile object
e2	ChIPprofile object
x	objects.
...	further arguments passed to methods.
base	a positive or complex number: the base with respect to which logarithms are computed. Defaults to $e=\exp(1)$ .

**Value**

A ChIPprofile object of result of arithmetic operation.

**Examples**

```
data(chipExampleBig)
chipExampleBig[[1]] + chipExampleBig[[2]]
```

---

orientBy

*Set strand by overlapping or nearest anchor GRanges*


---

**Description**

Set strand by overlapping or nearest anchor GRanges

**Usage**

```
orientBy(testRanges, anchorRanges)
```

**Arguments**

testRanges	The GRanges object to anchor.
anchorRanges	A GRanges object by which to anchor strand orientation.

**Value**

newRanges A GRanges object.

**Examples**

```
data(ik_Example)
strand(ik_Example[[1]]) <- "+"
anchoredGRanges <- orientBy(ik_Example[[2]], ik_Example[[1]])
```

---

plotRegion	<i>Plot regions</i>
------------	---------------------

---

### Description

A function to plot regions

### Usage

```
## S4 method for signature 'ChIPprofile'
plotRegion(object,
  gts, sampleData, groupData, summariseBy,
  colourBy, lineBy, groupBy,
  plotregion, outliers, freeScale)

## S4 method for signature 'ChIPprofile'
plotRegion(object = "ChIPprofile", gts = NULL,
  sampleData = NULL, groupData = NULL, summariseBy = NULL,
  colourBy = NULL, lineBy = NULL, groupBy = NULL, plotregion = "full",
  outliers = NULL, freeScale = FALSE)
```

### Arguments

object	A ChIPprofile object
gts	A list of character vectors or GRangesList
plotregion	region to plot. For combined plots with style "region", may be "start" or "end" to show full resolution of plot of edges.
groupData	Dataframe of metadata for groups
sampleData	Dataframe of metadata for sample
summariseBy	Column names from GRanges elementmetadata. Formula or character vector of column names to use to collapse genomic ranges to summarised profiles. summariseBy can not be used in conjunction with groups specified by gts argument.
colourBy	Character vector or formula of either column names from colData(object) containing sample metadata or character vector "group" to colour by groups in gts
lineBy	Character vector or formula of either column names from colData(object) containing sample metadata or character vector "group" to set line type by groups in gts
groupBy	Character vector or formula of either column names from colData(object) containing sample metadata or character "group" to colour by groups in gts
outliers	A numeric vector of length 1 containing proportion from limits to windsorise.]
freeScale	TRUE or FALSE to set whether ggplot 2 facets have their own scales. Useful for comparing multiple samples of differing depths without normalisation. Default is FALSE.

### Value

A gg object from ggplot2

**Author(s)**

Thomas Carroll

**Examples**

```
data(chipExampleBig)
plotRegion(chipExampleBig[[2]])
```

---

pwmCov

*Example motif coverage*

---

**Description**

This dataset contains an rlelist of motif coverage

**Usage**

```
data(pwmCov)
```

**Details**

- pwmCov

**Value**

A rlelist of motif coverage

---

pwmToCoverage

*PWM hits and motif scores as an RLElist*

---

**Description**

Creates rlelist of pwm hits.

Motif score as an RLElist

**Usage**

```
pwmToCoverage(pwm, genome, min = "70%", removeRand = FALSE,
  chrsOfInterest = NULL)
```

```
makeMotifScoreRle(pwm, regions, genome, extend, removeRand = FALSE,
  strandScore = "mean", atCentre = FALSE)
```

**Arguments**

pwm	A PWM matrix object.
genome	A BSgenome object
min	pwm score (as percentage of maximum score) cutoff
removeRand	Remove contigs with rand string
chrsOfInterest	Chromosomes to use
regions	GRanges object to include in pwm rlelist
extend	bps to extend regions by
strandScore	Method for averaging strand. Options are max, mean, sum, bothstrands
atCentre	TRUE/FALSE. TRUE assigns score onto 1bp position at centre of motif. FALSE assigns every basepair the sum of scores of all overlapping motifs.

**Value**

A RLElist of motif density per base pair to be used as input to main soggi function.

**Author(s)**

Thomas Carroll

**Examples**

```
data(pwmCov)
data(singleGRange)
```

---

singleGRange

*A single GRange*

---

**Description**

This dataset contains an rlelist of motif coverage

**Usage**

```
data(singleGRange)
```

**Details**

- singleGRange

**Value**

A single GRanges used in motif coverage example/

# Index

- \* **datasets**
  - chipExampleBig, [3](#)
  - ik\_Example, [7](#)
  - ik\_Profiles, [7](#)
  - pwmCov, [12](#)
  - singleGRange, [13](#)
- [[,ChIPprofile,ANY,missing-method
  - (c,ChIPprofile-method), [2](#)
- \$,ChIPprofile-method
  - (c,ChIPprofile-method), [2](#)
- as.integer, [3](#)
- backtick, [3](#)
  
- c,ChIPprofile-method, [2](#)
- cbind, [2](#)
- cbind,ChIPprofile-method
  - (c,ChIPprofile-method), [2](#)
- chipExampleBig, [3](#)
- ChIPprofile (ChIPprofile-class), [4](#)
- ChIPprofile-ChIPprofile
  - (ChIPprofile-class), [4](#)
- ChIPprofile-class, [4](#)
  
- dimnames, [3](#)
  
- findconsensusRegions, [5](#)
  
- groupByOverlaps, [6](#)
  
- ik\_Example, [7](#)
- ik\_Profiles, [7](#)
  
- log,ChIPprofile-method
  - (Ops,ChIPprofile,ChIPprofile-method), [9](#)
- log2,ChIPprofile-method
  - (Ops,ChIPprofile,ChIPprofile-method), [9](#)
  
- makeMotifScoreRle (pwmToCoverage), [12](#)
- mean,ChIPprofile-method
  - (Ops,ChIPprofile,ChIPprofile-method), [9](#)
  
- name, [3](#)
- names, [3](#)
- normalise, [8](#)
- normalise,ChIPprofile,character,numeric-method
  - (normalise), [8](#)
- normalise,ChIPprofile-method
  - (normalise), [8](#)
- normalise.ChIPprofile (normalise), [8](#)
- normaliseQuantiles, [8](#)
- normaliseQuantiles,ChIPprofile-method
  - (normaliseQuantiles), [8](#)
- normaliseQuantiles.ChIPprofile
  - (normaliseQuantiles), [8](#)
  
- Ops,ChIPprofile,ChIPprofile-method, [9](#)
- Ops,ChIPprofile,numeric-method
  - (Ops,ChIPprofile,ChIPprofile-method), [9](#)
- Ops,numeric,ChIPprofile-method
  - (Ops,ChIPprofile,ChIPprofile-method), [9](#)
  
- orientBy, [10](#)
  
- plotRegion, [11](#)
- plotRegion,ChIPprofile-method
  - (plotRegion), [11](#)
- plotRegion.ChIPprofile (plotRegion), [11](#)
- pwmCov, [12](#)
- pwmToCoverage, [12](#)
  
- rbind,ChIPprofile-method
  - (c,ChIPprofile-method), [2](#)
- regionPlot (ChIPprofile-class), [4](#)
  
- singleGRange, [13](#)
- soggi (ChIPprofile-class), [4](#)
- submitPipeline (findconsensusRegions), [5](#)