Package ‘chipseq’

November 20, 2016

Title chipseq: A package for analyzing chipseq data

Version 1.24.0

Author Deepayan Sarkar, Robert Gentleman, Michael Lawrence, Zizhen Yao

Description Tools for helping process short read data for chipseq experiments

Depends R (>= 2.10), methods, BiocGenerics (>= 0.1.0), S4Vectors (>= 0.9.25), IRanges (>= 1.99.1), GenomicRanges (>= 1.17.7), ShortRead

Imports methods, stats, lattice, BiocGenerics, IRanges, GenomicRanges, ShortRead

Suggests BSgenome, GenomicFeatures, TxDb.Musculus.UCSC.mm9.knownGene

Maintainer Bioconductor Package Maintainer

License Artistic-2.0

LazyLoad yes

biocViews ChIPSeq, Sequencing, Coverage, QualityControl, DataImport

NeedsCompilation yes

\textbf{\texttt{R} topics documented:}

\begin{verbatim}
chipseqFilter .................................................. 2
covrageplot .................................................... 3
cstest ........................................................... 3
diffPeakSummary ................................................ 4
estimate.mean.fraglen ......................................... 5
islandDepthPlot ............................................... 7
laneSubsample ............................................... 7
peakCutoff .................................................... 8
peakSummary-methods ......................................... 9
subsetSummary ................................................ 9
\end{verbatim}

Index 11
chipseqFilter  Filtering ChIP-seq reads

Description
Convenience for creating an SRFFilter object appropriate for ChIP-seq data. Typically, the result is passed to readAligned when loading reads.

Usage
chipseqFilter(exclude = "[_MXY]", uniqueness = c("location", "sequence", "location*sequence", "none"), hasStrand = TRUE)

Arguments
- **exclude**
  A regular expression for excluding chromosomes by name. Just like the parameter to bsapply.

- **uniqueness**
  The criteria used to determine whether a read is unique. A read may be unique if it maps to a unique location, has a unique sequence or both. Specifying none avoids this test entirely.

- **hasStrand**
  Whether to require that the read is mapped to a strand, which usually translates to whether the read was mapped at all.

Value
An SRFFilter object

Author(s)
M. Lawrence

Examples
```
sp <- SolexaPath(system.file("extdata", package="ShortRead"))
filter <- chipseqFilter()
aln <- readAligned(sp, "s_2_export.txt", filter=filter)
## allow mapping to the same location (but only if sequence is different)
filter <- chipseqFilter(uniqueness = "sequence")
aln <- readAligned(sp, "s_2_export.txt", filter=filter)
## allow sex chromosomes
filter <- chipseqFilter(exclude = "[M_]")
aln <- readAligned(sp, "s_2_export.txt", filter=filter)
```
**coverageplot**

*Plot coverage on a small interval.*

**Description**

A function that plots one or two coverage vectors over a relatively small interval in the genome.

**Usage**

```r
coverageplot(peaks1, peaks2 = NULL, i = 1,
    xlab = "Position", ylab = "Coverage",
    opposite = TRUE, ...)
```

**Arguments**

- **peaks1, peaks2** A set of peaks as described by ranges over a coverage vector.
- **i** Which peak to use.
- **xlab, ylab** Axis labels.
- **opposite** Logical specifying whether the two peaks should be plotted on opposite sides (appropriate for positive and negative strand peaks).
- **...** Extra arguments.

**Author(s)**

Deepayan Sarkar

**Examples**

```r
cov <- Rle(c(1:10, seq(10, 1, -2), seq(1,5,2), 4:1), rep(1:2, 11))
peaks <- slice(cov, 3)
peaks.cov <- Views(cov, ranges(peaks))
peaks.cov.rev <- rev(peaks.cov)
coverageplot(peaks.cov, peaks.cov.rev, ylab = "Example")
```

---

**cstest**

*A test ChIP-Seq dataset*

**Description**

A small subset of a ChIP-Seq dataset downloaded from the Short-Read Archive.

**Usage**

```r
data(cstest)
```

**Format**

The dataset is on object of class `GRangesList` with read alignments from three chromosomes in two lanes representing CTCF and GFP pull-down in mouse.
diffPeakSummary

Source


References


Examples

data(cstest)
names(cstest)
cstest$gfp

diffPeakSummary A function to identify and produce summary statistics for differentially expressed peaks.

Description

Given two sets of peaks, this function combines them and summarizes the individual coverage vectors under the combined peak set.

Usage

diffPeakSummary(ranges1, ranges2,
    viewSummary = list(sums = viewSums, maxs = viewMaxs))

Arguments

ranges1 First set of peaks (typically an RleViewsList).
ranges2 Second set of peaks (typically an RleViewsList).
viewSummary A list of the per peak summary functions.

Value

A data.frame with one row for each peak in the combined data. The chromosome, start and stop nucleotide positions (+ strand) are given as are the summary statistics requested.

Author(s)

D. Sarkar
estimate.mean.fraglen

Examples

data(cstest)
library(BSgenome.Musculus.UCSC.mm9)
seqlevels(cstest) <- seqlevels(Musculus)
seqlengths(cstest) <- seqlengths(Musculus)
## find peaks
findPeaks <- function(reads) {
    reads.ext <- resize(reads, width = 200)
    slice(coverage(reads.ext), lower = 8)
}
peakSummary <- diffPeakSummary(findPeaks(cstest$gfp), findPeaks(cstest$ctcf))

estimate.mean.fraglen

Estimate summaries of the distribution of fragment lengths in a short-read experiment. The methods are designed for ChIP-Seq experiments and may not work well in data without peaks.

Description

estimate.mean.fraglen implements three methods for estimating mean fragment length. The other functions are related helper functions implementing various methods, but may be useful by themselves for diagnostic purposes. Many of these operations are potentially slow.

sparse.density is intended to be similar to density, but returns the results in a run-length encoded form. This is useful when long stretches of the range of the data have zero density.

Usage

estimate.mean.fraglen(x, method = c("SISSR", "coverage", "correlation"),
        ...
)
basesCovered(x, shift = seq(5, 300, 5), seqLen = 100, verbose = FALSE)
densityCorr(x, shift = seq(0, 500, 5), center = FALSE,
        width = seqLen *2L, seqLen=100L, maxDist = 500L, ...)
sparse.density(x, width = 50, kernel = "epanechnikov",
        from = start(rix)[1] - 10L,
        to = end(rix)[length(rix)] + 10L)

Arguments

x

For estimate.mean.fraglen, typically an AlignedRead or a GRanges object. Also supported but deprecated, as they do not have formal strand information: RangedData (with a "strand" column), or a list-like object with elements "+" and "-" representing locations of reads aligned to positive and negative strands (the values should be integers denoting the location where the first sequenced base matched.) Supported (but again, deprecated) list types include: RangesList, IntegerList or an ordinary R list.

For basesCovered and densityCorr, a list with elements "+" and "-" representing locations of reads aligned to positive and negative strands (the values should be integers denoting the location where the first sequenced base matched.) densityCorr has also come to support GRanges input directly.
For `sparse.density`, a numeric or integer vector for which density is to be computed.

**method**
Character string giving method to be used. `method = "SISSR"` implements the method described in Jothi et al (see References below). `method = "correlation"` implements the method described in Kharchenko et al (see References below), where the idea is to compute the density of tag start positions separately for each strand, and then determine the amount of shift that maximizes the correlation between these two densities. `method = "coverage"` computes the optimal shift for which the number of bases covered by any read is minimized.

**shift**
Integer vector giving amount of shifts to be tried when optimizing. The current algorithm simply evaluates all supplied values and reports the one giving minimum coverage or maximum correlation.

**seqLen**
For the "coverage" method, the assumed length of each read for computing the coverage. Typically the read length. This is added to the shift estimated by "coverage" and "correlation" to come up with the actual fragment length.

**verbose**
Logical specifying whether progress information should be printed during execution.

**center**
For the "correlation" method, whether the calculations should incorporate centering by the mean density. The default is not to do so; as the density is zero over most of the genome, this slightly improves efficiency at negligible loss in accuracy.

**width**
Half-bandwidth used in the computation. This needs to be specified as an integer, data-driven rules are not supported.

**kernel**
A character string giving the density kernel.

**from, to**
specifies range over which the density is to be computed.

**maxDist**
If distance to nearest neighbor is more than this, the position is discarded. This removes isolated points, which are not very informative.

... Extra arguments, passed on as appropriate to other functions.

**Details**
For the correlation method, the range over which densities are computed only cover the range of reads; that is, the beginning and end of chromosomes are excluded.

**Value**

`estimate.mean.fraglen` gives an estimate of the mean fragment length.

`basesCovered` and `densityCorr` give a vector of the corresponding objective function evaluated at the supplied values of `shift`.

`sparse.density` returns an object of class "Rle".

**Author(s)**
Deepayan Sarkar, Michael Lawrence

**References**

### islandDepthPlot

**Examples**

```r
data(cstest)
estimate.mean.fraglen(cstest["ctcf"], method = "coverage")
```

---

**Description**

Plots the distribution of island depths using points for the observed islands and a line for the Poisson estimate of the noise. Useful for choosing a depth corresponding to a desired FDR.

**Usage**

```r
islandDepthPlot(x, maxDepth = 20L)
```

**Arguments**

- `x`: A coverage object, e.g., `RleList`.
- `maxDepth`: The maximum depth to plot (there are usually some outliers).

**Author(s)**

D. Sarkar, M. Lawrence

**See Also**

`peakCutoff` for calculating a cutoff value for an FDR.

---

### laneSubsample

**Description**

Subsamples data from multiple lanes on a per-chromosome basis.

**Usage**

```r
laneSubsample(lane1, lane2, fudge = 0.05)
```

**Arguments**

- `lane1, lane2`: Two lanes of data, each of class "GRanges".
- `fudge`: A numeric fudge factor. For each chromosome, if the difference in the sizes relative to the size of the first dataset is less than `fudge`, no subsampling is done.
Value

laneSubsample returns a list similar to its input, but with the larger dataset subsampled to be similar to the smaller one.

Author(s)

D. Sarkar

Examples

data(cstest)
## subsample to compare lanes
cstest.sub <- laneSubsample(cstest[[1]], cstest[[2]])
unlist(cstest.sub)

Description

Calculates a peak cutoff value given an FDR, assuming a Poisson noise distribution estimated from the frequency of singleton and doubleton islands.

Usage

peakCutoff(cov, fdr.cutoff = 0.001, k = 2:20)

Arguments

cov The coverage object, e.g., an RleList object.
fdr.cutoff The maximum-allowed FDR for calculating the cutoff.
k The coverage levels at which to estimate an FDR value. The maximal value that is less than fdr.cutoff is chosen for calculating the cutoff. Usually best left to the default.

Value

A numeric value to use for calling peaks

Author(s)

D. Sarkar and M. Lawrence

See Also

islandDepthPlot for the graphical equivalent; the vignette for a bit more explanation.

Examples

data(cstest)
cov <- coverage(resize(cstest$ctcf, width=200))
peakCutoff(cov)
peakSummary-methods  
**Summarizing peak sets**

**Description**

Summarizes a set of peaks into a `RangedData` object with columns of statistics like the peak maxima and integrals (sums).

**Usage**

```r
peakSummary(x, ...)
```

**Arguments**

- `x`: An object containing peaks, usually a `RleViewsList`.
- `...`: Arguments to pass to methods

**Value**

A `RangedData` object of the peaks, with columns named `max`, `maxpos` (position of the maximum, centered), and `sum`.

**See Also**

`view-summarization-methods` in the `IRanges` package for view summarization methods like `viewMaxs` and `viewSums`.

---

subsetSummary  
**Compute summaries for cumulative subsets of a short-read data set.**

**Description**

**THIS FUNCTION IS DEFUNCT!**

Divides a short-read dataset into several subsets, and computes various summaries cumulatively. The goal is to study the characteristics of the data as a function of sample size.

**Usage**

```r
subsetSummary(x, chr, nstep, props = seq(0.1, 1, 0.1),
             chromlens = seqlengths(x), fg.cutoff = 6, seqLen = 200,
             fdr.cutoff = 0.001, use.fdr = FALSE, resample = TRUE,
             islands = TRUE, verbose = getOption("verbose"))
```
Arguments

x A "GRanges" object representing alignment locations at the sample level.
chr The chromosome for which the summaries are to be obtained. Must specify a valid element of x
nstep The number of maps in each increment for the full dataset (not per-chromosome). This will be translated to a per-chromosome number proportionally.
props Alternatively, an increasing sequence of proportions determining the size of each subset. Overrides nstep.
chromlens A named vector of per-chromosome lengths, typically the result of seqlengths.
fg.cutoff The coverage depth above which a region would be considered foreground.
seqLen The number of bases to which to extend each read before computing coverage.
resample Logical; whether to randomly reorder the reads before dividing them up into subsets. Useful to remove potential order effects (for example, if data from two lanes were combined to produce x).
fdr.cutoff The maximum false discovery rate for a region that is considered to be foreground.
use.fdr Whether to use the FDR detected peaks when calling foreground and background.
islands Logical. If TRUE, the whole island would be considered foreground if the maximum depth equals or exceeds fg.cutoff. If FALSE, only the region above the cutoff would be considered foreground.
verbose logical controlling whether progress information will be shown during computation (which is potentially long-running).

Value

A data frame with various per-subset summaries.

Note

This function should be considered preliminary, in that it might change significantly or simply be removed in a subsequent version. If you like it the way it is, please notify the maintainer.

Author(s)

Deepayan Sarkar, Michael Lawrence
Index

*Topic **datasets**
cstest, 3
*Topic **hplot**
coverageplot, 3
*Topic **manip**
laneSubsample, 7
*Topic **methods**
peakSummary-methods, 9
*Topic **univar**
estimate.mean.fraglen, 5
subsetSummary, 9
*Topic **utilities**
laneSubsample, 7

AlignedRead, 5
basesCovered (estimate.mean.fraglen), 5
bsapply, 2
chipseqFilter, 2
covrageplot, 3
cstest, 3
density, 5
densityCorr (estimate.mean.fraglen), 5
densityCorr, GenomicRanges
(estimate.mean.fraglen), 5
densityCorr, list
(estimate.mean.fraglen), 5
diffPeakSummary, 4
diffPeakSummary,RleViewsList,RleViewsList-method
(diffPeakSummary), 4

estimate.mean.fraglen, 5
estimate.mean.fraglen,AlignedRead-method
(estimate.mean.fraglen), 5
estimate.mean.fraglen,GRanges-method
(estimate.mean.fraglen), 5

GRanges, 5
IntegerList, 5
islandDepthPlot, 7, 8

laneSubsample, 7

peakCutoff, 7, 8
peakSummary (peakSummary-methods), 9
peakSummary,RleViews-method
(peakSummary-methods), 9
peakSummary,RleViewsList-method
(peakSummary-methods), 9
peakSummary-methods, 9
RangedData, 5, 9
RangesList, 5
readAligned, 2
RleList, 7, 8
RleViewsList, 4, 9
seqlengths, 10
sparse.density (estimate.mean.fraglen), 5
SRFilter, 2
subsetSummary, 9
view-summarization-methods, 9