Package: Basic4Cseq (via r-universe)

November 1, 2026

Type Package

Title Basic4Cseq: an R/Bioconductor package for analyzing 4C-seq data

Version 1.48.0

Date 2025-06-27

Author Carolin Walter

Maintainer Carolin Walter <carolin.walter@uni-muenster.de>

Imports methods, RCircos, BSgenome.Ecoli.NCBI.20080805

Depends R (>= 3.4), Biostrings, GenomicAlignments, caTools, GenomicRanges, grDevices, graphics, stats, utils

Suggests BSgenome.Hsapiens.UCSC.hg19

Description Basic4Cseq is an R/Bioconductor package for basic filtering, analysis and subsequent visualization of 4C-seq data. Virtual fragment libraries can be created for any BSGenome package, and filter functions for both reads and fragments and basic quality controls are included. Fragment data in the vicinity of the experiment's viewpoint can be visualized as a coverage plot based on a running median approach and a multi-scale contact profile.

License LGPL-3

biocViews ImmunoOncology, Visualization, QualityControl, Sequencing, Coverage, Alignment, RNASeq, SequenceMatching, DataImport

Config/pak/sysreqs make libbz2-dev liblzma-dev libxml2-dev libssl-dev xz-utils zlib1g-dev

Repository Bioconductor 3.23

Date/Publication 2026-04-28 12:38:43 UTC

RemoteUrl https://github.com/bioc/Basic4Cseq

RemoteRef RELEASE_3_23

RemoteSha 3c243e7af13d3775d57bb86fb9025f357cdc1e6f

Contents

checkRestrictionEnzymeSequence . . . 2
chooseNearCisFragments . . . 3
createVirtualFragmentLibrary . . . 4
Data4Cseq . . . 6
Data4Cseq-class . . . 7
drawDigestionFragmentHistogram . . . 9
drawHeatmap . . . 10
exportVisualizationFragmentData . . . 11
getReadDistribution . . . 12
giveEnzymeSequence . . . 13
importVisualizationFragmentData . . . 14
liverData . . . 14
liverDataRaw . . . 15
normalizeFragmentData . . . 16
plotTransInteractions . . . 16
prepare4CseqData . . . 18
printBEDFragmentLibrary . . . 19
printWigFile . . . 20
readPointsOfInterestFile . . . 21
readsToFragments . . . 22
simulateDigestion . . . 22
visualizeViewpoint . . . 23

Index **25**

`checkRestrictionEnzymeSequence`

Remove invalid 4C-seq reads from a SAM file

Description

Basic4Cseq offers filter functions for invalid 4C-seq reads. This function removes 4C-seq reads from a provided Sequence Alignment/Map (SAM) file that show mismatches in the restriction enzyme sequence.

Usage

```
checkRestrictionEnzymeSequence(firstCutter, inputFileName, outputFileName = "output.sam", keepOnlyUni
```

Arguments

`firstCutter`	First restriction enzyme sequence of the 4C-seq experiment
`inputFileName`	Name of the input SAM file that contains aligned reads for the 4C-seq experiment
`outputFileName`	Name of the output SAM file that is created to store the filtered 4C-seq reads

keepOnlyUniqueReads
: If TRUE, delete non-unique reads. Information in the SAM flag field is used to determine whether a read is unique or not.

writeStatistics
: If TRUE, write statistics (e.g. the number of unique reads) to a text file

Details

Valid 4C-seq reads start at a primary restriction site and continue with its downstream sequence, so any mismatch in the restriction enzyme sequence of a read is an indicator for a mismatch. The mapping information of the restriction enzyme sequence bases of a read (if present) can be used for filtering purposes. checkRestrictionEnzymeSequence tests the first bases of a read (depending on the length of the first restriction enzyme either 4 or 6 bp long) for mismatches. Reads with mismatches in the restriction enzyme sequence are deleted, the filtered data is then written to a new SAM file. The function does not yet differentiate between blind and nonblind fragments, but removes potential misalignments that may overlap with valid fragment ends and distort the true 4C-seq signal.

Value

A SAM file containing the filtered valid 4C-seq reads

Note

The use of the function is only possible if the restriction enzyme sequence is not trimmed or otherwise absent.

Author(s)

Carolin Walter

Examples

```
if(interactive()) {
    file <- system.file("extdata", "fetalLiverCutter.sam", package="Basic4Cseq")
    checkRestrictionEnzymeSequence("aagctt", file)
}
```

---

chooseNearCisFragments

Choose fragments in a provided region around the viewpoint

Description

This function extracts fragment data from a Data4Cseq object's rawFragments slot for visualization with the functions visualizeViewpoint and drawHeatmap . Relevant fragments are located within the chosen visualization range; the viewpoint itself can be excluded or included.

Usage

```
chooseNearCisFragments(expData, regionCoordinates, deleteViewpoint = TRUE)
```

Arguments

expData	Experiment data of class Data4Cseq with information on the 4C-seq experiment, including fragment data for the viewpoint chromosome
regionCoordinates	Interval on the viewpoint chromosome for the intended visualization
deleteViewpoint	If TRUE, delete all fragments that intersect with the experiment's viewpoint interval

Value

A data frame containing the chosen near-cis fragments

Note

Viewpoint fragments are removed per default to prevent bias through overrepresented sequences caused by self-ligation. These fragments can be included, but should be interpreted with caution.

Author(s)

Carolin Walter

Examples

```
 # read example data
 data(liverData)
fragments<-chooseNearCisFragments(liverData, regionCoordinates = c(20800000, 21000000))
 head(fragments)
```

---

createVirtualFragmentLibrary

Create a virtual fragment library from a provided genome and two restriction enzymes

---

Description

Basic4Cseq can create virtual fragment libraries from any BSgenome package or DNAString object. Two restriction enzymes have to be specified to cut the DNA, the read length is needed to check the fragment ends of corresponding length for uniqueness. Filter options (minimum and maximum size) are provided on fragment level and on fragment end level.

Usage

```
createVirtualFragmentLibrary(chosenGenome, firstCutter, secondCutter, readLength, onlyNonBlind = TRUE
```

Arguments

chosenGenome	The genome that is to be digested in silico with the provided enzymes; can be an instance of BSgenome or DNAString
firstCutter	First of two restriction enzymes
secondCutter	Second of two restriction enzymes
readLength	Read length for the experiment
onlyNonBlind	Variable that is TRUE (default) if only non-blind fragments are considered (i.e. all blind fragments are removed)
useOnlyIndex	Convenience function to adapt the annotation style of the chromosomes ("chr1", ..."chrY" or "1", ..., "Y"); parameter has to be set to match the BAM file in question
minSize	Filter option that allows to delete fragments below a certain size (in bp)
maxSize	Filter option that allows to delete fragments above a certain size (in bp)
minFragEndSize	Filter option that allows to delete fragment ends below a certain size (in bp)
maxFragEndSize	Filter option that allows to delete fragment ends above a certain size (in bp)
useAllData	Variable that indicates if all data of a BSgenome package is to be used. If FALSE, chromosome names including a "_" are removed, reducing the set of chromosomes to (1 ... 19, X, Y, MT) for the mouse genome or (1 ... 22, X, Y, MT) for the human genome
chromosomeName	Chromosome name for the virtual fragment library if a DNAString object is used instead of a BSgenome object.
libraryName	Name of the file the created virtual fragment library is written to. Per default the file is called "fragments_firstCutter_secondCutter.csv". The fragment data is returned as a data frame if and only if an empty character string is chosen as libraryName.

Details

- readLength is relevant for the creation of the virtual fragment library to differenciate between unique and non-unique fragment ends. While two fragments can be unique, their respective ends may be repetitive if only the first few bases are considered. For 4C-seq data, reads can only map to the start (or end, respectively) of a 4C-seq fragment, the remaining fragment part is not covered. The length of a fragment end that has to be checked for uniqueness therefore depends on the read length of the experiment.
- useAllData uses the lengths of the chromosomes to identify relevant ones, based on the current BSgenome packages for mm10 or hg19, and may therefore provide undesirable results for smaller genomes with different lengths (i.e. discard all chromosomes).
- The length of a fragment influences the expected read count of a 4C-seq fragment. Per default, **Basic4Cseq** uses the experiment's read length as minimum fragment end size and places virtually no limit on the maximum fragment end size.

Value

A tab-separated file with the specified virtual fragment library (containing fragment position, length, presence of second restriction enzyme and uniqueness of the fragment ends)

Note

- It is strongly recommended to preprocess and store the virtual fragment library if a number of experiments with the same restriction enzyme combination, read length and underlying genome have to be analyzed.
- Processing one of the larger BSgenome packages takes some time and computer data storage.
- If no library name for the virtual fragment library is specified, the fragment data is returned as a data frame. If the library name "default" is chosen, the tab-separated file is named "fragments_firstCutter_secondCutter" (with variable cutter sequences).

Author(s)

Carolin Walter

Examples

```
if(interactive()) {
    library(BSgenome.Ecoli.NCBI.20080805)
 fragmentData = createVirtualFragmentLibrary(chosenGenome = Ecoli$NC_002655, firstCutter = "catg", secondCutt
}
```

Data4Cseq *Creating a Data4Cseq object*

Description

This function creates a Data4Cseq object. Data on the 4C-seq experiment, e.g. the chromosome of the viewpoint, is stored and checked for consistency.

Usage

```
Data4Cseq(viewpointChromosome, viewpointInterval, readLength, pointsOfInterest, rawReads)
```

Arguments

`viewpointChromosome`	The experiment's viewpoint chromosome
`viewpointInterval`	The interval of the experiment's viewpoint, consisting of a start and end coordinate
`readLength`	The experiment's read length (in base pairs)
`pointsOfInterest`	Points of interest to be marked in a near-cis visualization
`rawReads`	Reads of the 4C-seq experiment, aligned and stored as an `GAlignments` object

Details

A Data4Cseq object contains basic information for the corresponding 4C-seq experiment, including the viewpoint chromosome, the viewpoint region and reads from the experiment. See Data4Cseq-class for more details. The constructor collects the basic data; fragment data or normalized read counts are added later.

Fragments at the experiment's viewpoint are usually vastly overrepresented due to self-ligation; chooseNearCisFragments offers the option to discard all fragments in the specified viewpoint region. The specified viewpoint interval of a Data4Cseq object is supposed to correspond to the positions of the biological primers on the genome, but can also be increased in size if more fragments around the viewpoint should be removed.

Value

An instance of the Data4Cseq class.

Author(s)

Carolin Walter

See Also

Data4Cseq-class

Examples

```
  # create a Data4Cseq object with a minimum of data
liverData = Data4Cseq(viewpointChromosome = "10", viewpointInterval = c(20879870, 20882209), readLength = 54)
  liverData

  # create a Data4Cseq object, including possible points of interest and raw reads
  bamFile <- system.file("extdata", "fetalLiverShort.bam", package="Basic4Cseq")
  liverReads <- readGAlignments(bamFile)
pointsOfInterestFile <- system.file("extdata", "fetalLiverVP.bed", package="Basic4Cseq")
liverData = Data4Cseq(viewpointChromosome = "10", viewpointInterval = c(20879870, 20882209), readLength = 54, po
  liverData
```

Data4Cseq-class *Class* "Data4Cseq"

Description

This class is a container for information on a specific 4C-seq experiment. Stored information includes raw reads, fragment data, and the experiment's viewpoint location.

Objects from the Class

Objects can be created by calls of the form new("Data4Cseq", ...).

Slots

`viewpointChromosome`: Object of class `"character"` representing the viewpoint chromosome's name

`viewpointInterval`: Object of class `"numeric"` representing the viewpoint interval's location

`readLength`: Object of class `"numeric"` representing the experiment's read length

`pointsOfInterest`: Object of class `"data.frame"` representing any points of interest to be marked in the near-cis visualizations

`rawReads`: Object of class `"GAlignments"` representing the raw 4C-seq reads of the experiment

`rawFragments`: Object of class `"data.frame"` representing the experiment's corresponding virtual fragment library

`nearCisFragments`: Object of class `"data.frame"` representing near-cis data in fragment form

Methods

viewpointChromosome<- `signature(object = "Data4Cseq", value = "character")`: Setter-method for the viewpointChromosome slot.

viewpointChromosome `signature(object = "Data4Cseq")`: Getter-method for the viewpointChromosome slot.

viewpointInterval<- `signature(object = "Data4Cseq", value = "numeric")`: Setter-method for the viewpointInterval slot.

viewpointInterval `signature(object = "Data4Cseq")`: Getter-method for the viewpointInterval slot.

readLength<- `signature(object = "Data4Cseq", value = "numeric")`: Setter-method for the readLength slot.

readLength `signature(object = "Data4Cseq")`: Getter-method for the readLength slot.

pointsOfInterest<- `signature(object = "Data4Cseq", value = "data.frame")`: Setter-method for the pointsOfInterest slot.

pointsOfInterest `signature(object = "Data4Cseq")`: Getter-method for the pointsOfInterest slot.

rawReads<- `signature(object = "Data4Cseq", value = "GAlignments")`: Setter-method for the rawReads slot.

rawReads `signature(object = "Data4Cseq")`: Getter-method for the rawReads slot.

rawFragments<- `signature(object = "Data4Cseq", value = "data.frame")`: Setter-method for the rawFragments slot.

rawFragments `signature(object = "Data4Cseq")`: Getter-method for the rawFragments slot.

nearCisFragments<- `signature(object = "Data4Cseq", value = "data.frame")`: Setter-method for the nearCisFragments slot.

nearCisFragments `signature(object = "Data4Cseq")`: Getter-method for the nearCisFragments slot.

Author(s)

Carolin Walter

Examples

```
showClass("Data4Cseq")
```

`drawDigestionFragmentHistogram`
Visualize digestion fragments with a histogram

Description

This function is a small convenience function to plot the results of `simulateDigestion` as a histogram. Minimum and maximum fragment lengths can be specified to visualize a specified interval of the fragment data.

Usage

```
drawDigestionFragmentHistogram(fragments, minLength = 0, maxLength = 10000)
```

Arguments

`fragments`	Fragment data to visualize (data frame with lengths and corresponding frequencies)
`minLength`	Minimum fragment length to visualize
`maxLength`	Maximum fragment length to visualize

Value

Histogram plot of the fragment data

Author(s)

Carolin Walter

Examples

```
shortTestGenome = "ATCCATGTAGGCTAAGTACACATGTTAAGGTACAGTACAATTGCACGATCAT"
fragments = simulateDigestion("catg", "gtac", shortTestGenome)
drawDigestionFragmentHistogram(fragments)
```

drawHeatmap *Draw a heatmap-like multi-scale contact profile*

Description

This method draws a fragment-based heatmap-like plot for 4C-seq data around a given viewpoint. For a given number of bands, color-coded running medians or running means of signal intensity (normalized and log-scaled) in different fragments are displayed; the window size of the running medians or running means increases from top to bottom. A corresponding colour legend is added in an extra plot.

Usage

```
    ## S4 method for signature 'Data4Cseq'
drawHeatmap(expData, plotFileName = "", smoothingType = "median", picDim = c(9, 2.2), bands = 5, cutoffL
    ## S4 method for signature 'data.frame'
drawHeatmap(expData, plotFileName = "", smoothingType = "median", picDim = c(9, 2.2), bands = 5, cutoffL
```

Arguments

expData	Experiment data from a given 4C-seq experiment for visualization; can be a Data4Cseq object or a data frame
plotFileName	Name for the heatmap plot
smoothingType	Type of interpolation (running mean or running median). Default value is "median" (i.e. running median)
picDim	Dimensions of the plot. Default value is c(9, 2.2), to fit a small heatmap plot below the main 4C-seq plot that is created by visualizeViewpoint
bands	Number of coloured "bands" (rows) to visualize. The first band contains the raw data (running median or running mean with window size 1), the following bands increase in window size (+2 per band)
cutoffLog	Cut off value for the logarithmic scale
xAxisIntervalLength	Length of the x axis intervals in the plot
legendLabels	Labels for a heat colour legend plot; labels should correspond to the logarithmic cut offs
useFragEnds	Indicates whether fragment end data is used directly or interpolated on fragment level

Value

A multiscale intensity contact profile plot and a corresponding colour legend)

Note

PDF export and output as TIFF format are supported. The export format is chosen depending on the plot file name's ending. If no plot file name is provided, the result is plotted on screen.

Author(s)

Carolin Walter

Examples

```
if(interactive()) {
    data(liverData)
    drawHeatmap(liverData)
}
```

---

exportVisualizationFragmentData

Export near-cis fragment data of a `Data4Cseq` *object*

Description

This function is a simple helper function that writes the near-cis data of a `Data4Cseq` object as tab-separated file to hard disk.

Usage

```
exportVisualizationFragmentData(expData, fileName, fullData = FALSE)
```

Arguments

`expData`	Experiment data of class `Data4Cseq` information on the 4C-seq experiment, including visualization data
`fileName`	Name for the tab-separated file
`fullData`	If TRUE, the function exports the full fragment data (including fragment end length etc). If FALSE, only the minimum fragment information is exported, i.e. chromosome, start, end and (normalized) read count.

Value

A tab-separated file containing near-cis framgent data of a `Data4Cseq` object

Author(s)

Carolin Walter

Examples

```
if(interactive()) {
    data(liverData)
    exportVisualizationFragmentData(liverData, "fetalLiverData.csv")
}
```

`getReadDistribution` *Calculate the read distribution for a 4C-seq experiment*

Description

This function provides some 4C-seq quality statistics based on the experiment's read distribution throughout the genome. `getReadDistribution` calculates the number of total reads, cis to overall ratio of reads, and the percentage of covered fragment ends within a certain distance around the experiment's viewpoint. Reference values for high-quality experiments, as provided by van de Werken et al, 2012, are more than one million reads total, a cis to overall ratio of more than 40% and a large fraction of covered fragment ends in the viewpoint's vicinity.

Usage

```
getReadDistribution(expData, distanceFromVP = 100000, useFragEnds = TRUE, outputName = "")
```

Arguments

`expData`	Experiment data of class `Data4Cseq` with information on the 4C-seq experiment
`distanceFromVP`	Distance from the viewpoint that is checked for covered fragments
`useFragEnds`	If TRUE, the function uses fragment end data; if FALSE, an average value for whole fragments is used.
`outputName`	An optional name for an output text file containing the statistics data

Value

Text with statistics data on the 4C-seq experiment

Note

Text export is supported; if no file name is provided, the results are printed on screen.

Author(s)

Carolin Walter

References

van de Werken, H., de Vree, P., Splinter, E., et al. (2012): 4C technology: protocols and data analysis, Methods Enzymology, 513, 89-112

Examples

```
data(liverData)
getReadDistribution(liverData)
```

giveEnzymeSequence *Provide the corresponding enzyme sequence for an enzyme name*

Description

This function is a small convenience function that reads in a prepared file with restriction enzyme sequence names and sequences. giveEnzymeSequence then provides restriction enzyme sequences for the example enzymes listed in van de Werken et al's 4Cseqpipe data base.

Usage

```
giveEnzymeSequence(fileNameDatabase, enzymeName)
```

Arguments

fileNameDatabase	File name of the prepared enzyme database
enzymeName	Name of the enzyme for which the sequence is to be returned

Value

Character string with the restriction enzyme sequence

Note

For any custom-made enzyme list it is assumed that there are no duplicate enzyme names in the database.

Author(s)

Carolin Walter

References

van de Werken, H., Landan, G., Holwerda, S., et al. (2012): Robust 4C-seq data analysis to screen for regulatory DNA interactions, Nature Methods, 9, 969-971.

Examples

```
file <- system.file("extdata", "enzymeData.csv", package="Basic4Cseq")
giveEnzymeSequence(file, "NlaIII")
```

importVisualizationFragmentData *Import visualization data from a file*

Description

This function is a simple helper function that can import near-cis data which was previously exported and stored as tab-separated file.

Usage

```
importVisualizationFragmentData(fileName)
```

Arguments

fileName	Name for the tab-separated file with near-cis fragment data

Value

Data frame containing the near-cis fragment data

Author(s)

Carolin Walter

Examples

```
file <- system.file("extdata", "fetalLiver_finalFragments.csv", package="Basic4Cseq")
importVisualizationFragmentData(file)
head(file)
```

liverData *Example 4C-seq data set of fetal liver data*

Description

This data set contains an instance of a Data4Cseq object; 2185 reads on 453 fragments are included.

The 4C-seq data was taken from Stadhouders et al's fetal liver data set.

Usage

```
data("liverData")
```

Format

Formal class 'Data4Cseq'

Source

Shortened version of Stadhouders et al's fetal liver data:

Stadhouders, R., Thongjuea, S., et al. (2012): Dynamic long-range chromatin interactions control Myb proto-oncogene transcription during erythroid development. EMBO, 31, 986-999.

Examples

```
data("liverData")
liverData
```

---

liverDataRaw *Example 4C-seq data set of fetal liver data*

Description

This data set contains an instance of a `Data4Cseq` object; 2185 reads on 453 fragments are included. Raw reads are mapped to fragments, but the read count has not yet been normalized.

The 4C-seq data was taken from Stadhouders et al's fetal liver data set.

Usage

```
data("liverDataRaw")
```

Format

Formal class 'Data4Cseq'

Source

Shortened version of Stadhouders et al's fetal liver data:

Stadhouders, R., Thongjuea, S., et al. (2012): Dynamic long-range chromatin interactions control Myb proto-oncogene transcription during erythroid development. EMBO, 31, 986-999.

Examples

```
data("liverDataRaw")
liverDataRaw
```

normalizeFragmentData *Normalize near-cis fragment data read count*

Description

This function provides a simple RPM (reads per million) normalization for near-cis fragment data read counts of a `Data4Cseq` object. A form of normalization is especially important for the comparison of samples with a different read count.

Usage

```
normalizeFragmentData(expData)
```

Arguments

`expData`	Experiment data of class `Data4Cseq` with information on the 4C-seq experiment

Value

Data frame with RPM-normalized data

Author(s)

Carolin Walter

Examples

```
data(liverDataRaw)
normalizedFragments<-normalizeFragmentData(liverDataRaw)
head(normalizedFragments)
```

plotTransInteractions *Visualize trans interaction intervals*

Description

This function visualizes trans interaction intervals of a 4C-seq experiment with the help of the **RCircos** package. Significant interactions can be obtained by use of Splinter et al's `significant_interactions` code or similar algorithms.

Usage

```
plotTransInteractions(interactionFile, chromosomeViewpoint, coordViewpoint, ideogramData, PlotColor =
```

Arguments

interactionFile	Interaction interval data; either a file name or a data frame
chromosomeViewpoint	Viewpoint chromosome of the 4C-seq experiment
coordViewpoint	Viewpoint coordinates of the 4C-seq experiment
ideogramData	Ideogram data to be visualized in the RCirco-plot; either a file name or a data frame
PlotColor	Plot colours for the visualized interactions
expandBands	If TRUE, add a specified value to the size of the interaction intervals to increase the visibility of very small interactions
expansionValue	Value that is added to each interaction interval end
plotFileName	Optional name for an output file
picDim	Dimensions of the plot

Details

The code of Splinter et al to determine significant interactions provides chromosome, start and end of interaction intervals and a forth column with information on far-cis or trans data. This column is ignored by `plotTransInteractions`; it is assumed that all interactions for trans visualization are indeed trans interactions. Otherwise, far-cis interactions are visualized as well. While not a mistake per se, the (usually more numerous) far-cis interactions are easier to interpret if visualized with Splinter et al's spider-plot functions.

Value

An RCircos-plot of trans interaction intervals

Note

PDF export and output as TIFF format are supported. The export format is chosen depending on the plot file name's ending. If no plot file name is provided, the result is plotted on screen.

Author(s)

Carolin Walter

References

Zhang, H., Meltzer, P. and Davis, S. (2013) RCircos: an R package for Circos 2D track plots, BMC Bioinformatics, 14, 244

Splinter, E., de Wit, E., van de Werken, H., et al. (2012) Determining long-range chromatin interactions for selected genomic sites using 4C-seq technology: From fixation to computation, Methods, 58, 221-230.

Examples

```
if(interactive()) {
    library(RCircos)
 interactions <- system.file("extdata", "transInteractionData.txt", package="Basic4Cseq")
 ideograms <- system.file("extdata", "RCircos_GRCm38_ideogram.csv", package="Basic4Cseq")
 plotTransInteractions(interactions, "10", c(20000042, 20001000), ideograms, PlotColor = "blue", expandBands
}
```

`prepare4CseqData` *Alignment and filtering of raw 4C-seq data*

Description

This function is an optional wrapper for the alignment and preliminary filtering of 4C-seq data. `prepare4CseqData` reads a provided 4C-seq fastq file from hard disk. Alignment of the reads is done with BWA, the function `checkRestrictionEnzymeSequence` is used for optional filtering. Samtools and bedtools provide the necessary functionality for intersecting the filtered reads with a given 4C-seq fragment library for visualization purposes (e.g. with the Integrative Genomics Viewer, IGV).

Usage

```
prepare4CseqData(fastqFileName, firstCutter, fragmentLibrary, referenceGenome, pathToBWA = "", pathToS
```

Arguments

`fastqFileName`	The name of the fastq file that contains the 4C-seq reads
`firstCutter`	First cutting enzyme sequence for the 4C-seq experiment, e.g. "AAGCTT"
`fragmentLibrary`	Name of the fragment library to use for the current 4C-seq experiment; has to correspond to the chosen cutters and chosen genome
`referenceGenome`	Name (plus path) of the reference genome to use
`pathToBWA`	Path to BWA
`pathToSam`	Path to samtools
`pathToBED`	Path to bedtools
`controlCutterSequence`	If TRUE, the function `checkRestrictionEnzymeSequence` is used to filter non-valid 4C-seq reads
`bwaThreads`	Number of BWA threads
`minFragEndLength`	Minimum fragment end length to use for BED export

Value

computes and writes sorted .bam file for the data, as long as BWA, samtools and bedtools are available

Author(s)

Carolin Walter

References

Li, H. and Durbin, R. (2009) Fast and accurate short read alignment with Burrows-Wheeler Transform, Bioinformatics, 25, 1754-60.

Helga Thorvaldsdottir, James T. Robinson, Jill P. Mesirov. Integrative Genomics Viewer (IGV): high-performance genomics data visualization and exploration. Briefings in Bioinformatics 2012.

See Also

checkRestrictionEnzymeSequence

Examples

```
if(interactive()) {
    # BWA, samtools and bedtools must be installed
 # It is assumed that the example data files (from the package) are in the active directory
 prepare4CseqData("veryShortExample.fastq", "CATG", "veryShortLib.csv", referenceGenome = "veryShortReferenc
}
```

`printBEDFragmentLibrary` *Print a BED-file fragment library*

Description

This function extracts the first columns of a virtual fragment library file and exports them as a BED-file for use with other tools (e.g. visualization in the Integrative Genomics Viewer (IGV).)

Usage

```
printBEDFragmentLibrary(fragmentLibrary, BEDLibraryName, minFragEndLength = 0, zeroBased = FALSE)
```

Arguments

`fragmentLibrary`	Virtual fragment library file name
`BEDLibraryName`	File name for the exported BED file
`minFragEndLength`	Minimum fragment end length to be considered
`zeroBased`	If TRUE, adapt the start of the BED-file fragments

Value

writes BED-file containing the virtual fragment library position data

Author(s)

Carolin Walter

Examples

```
if(interactive()) {
 file <- system.file("extdata", "vfl_aagctt_catg_mm9_54_vp.csv", package="Basic4Cseq")
    printBEDFragmentLibrary(file, "BEDLibrary_FL_vp.bed")
}
```

---

printWigFile *Print a wig file from 4C-seq read data*

Description

This function provides wig files from filtered fragment data. Only reads on unique frag-ends are considered for the export. Export of wig files with a fixed span length requires a uniform read length throughout the data.

While some tools (e.g. the Integrative Genomics Viewer, IGV) accept 'raw' wig data, the UCSC browser needs a header line for correct visualizations. A basic header line has the form 'track type=wiggle_0', but may also contain information on the track's name and a short description. Since the header line may complicate possible downstream analysis of the wig files, no header is included per default.

Usage

```
printWigFile(expData, wigFileName = "output.wig", fixedSpan = TRUE, headerUCSC = "", useOnlyIndex = FAL
```

Arguments

expData	Experiment data of class Data4Cseq with information on the 4C-seq experiment
wigFileName	Name of the wig file that is written to hard disk
fixedSpan	If TRUE, use a fixed span for the wig file
headerUCSC	A header line for the UCSC browser
useOnlyIndex	If TRUE, use only '1,2,...Y' as chromosome names, if FALSE, use 'chr1,chr2...chrY'.

Value

A wig file containing the experiment's reads

Author(s)

Carolin Walter

References

UCSC Genome Browser: Kent WJ, Sugnet CW, Furey TS, Roskin KM, Pringle TH, Zahler AM, Haussler D. The human genome browser at UCSC. Genome Res. 2002 Jun;12(6):996-1006.

http://genome.ucsc.edu/

Examples

```
if(interactive()) {
    data(liverData)
    printWigFile(liverData, wigFileName = "fetalLiver.wig")
}
```

`readPointsOfInterestFile` *Read a file with coordinates of marker points*

Description

This small helper function reads a tab-separated file with points of interest information stored in a BED-like format. The file has to provide the columns "chromosome", "start", "end", "name" and "colour" of the regions. The data can then be used for marking the points in near-cis visualization plots, as provided by `visualizeViewpoint` and `drawHeatmap`.

Usage

```
readPointsOfInterestFile(poiFile)
```

Arguments

`poiFile`	Name of the input file (tab-separated)

Value

Data frame with information on points of interest for the near-cis visualizations

Author(s)

Carolin Walter

Examples

```
file <- system.file("extdata", "fetalLiverVP.bed", package="Basic4Cseq")
pointsOfInterests = readPointsOfInterestFile(file)
pointsOfInterests
```

readsToFragments *Determine fragment coverage of a 4C-seq fragment library*

Description

This function maps aligned reads to fragment ends of the virtual fragment library to calculate the coverage of the fragments. The number of reads at the start and end of a fragment is provided, as well as the average of both fragment ends.

Usage

```
readsToFragments(expData, fragmentLib)
```

Arguments

expData	Experiment data of class Data4Cseq with information on the 4C-seq experiment, including raw 4C-seq read data
fragmentLib	Fragment library for the given genome and cutting enzyme combination

Value

Data frame containing fragment-based data, i.e. a fragment's position and read coverage

Author(s)

Carolin Walter

Examples

```
data(liverData)
file <- system.file("extdata", "vfl_aagctt_catg_mm9_54_vp.csv", package="Basic4Cseq")
rawFragments(liverData) = readsToFragments(liverData, file)
head(rawFragments(liverData))
```

simulateDigestion *Simulate the digestion of a genome*

Description

This function simulates the digestion process with two restriction enzymes for a dna sequence or a BSgenome package. The information can then be used for quality controls of the biological fragment library.

Usage

```
simulateDigestion(firstCutter, secondCutter, dnaSequence)
```

Arguments

`firstCutter`	First restriction enzyme sequence for the digestion process
`secondCutter`	Second restriction enzyme sequence for the digestion process
`dnaSequence`	DNA sequence that is digested

Details

The resulting virtual library of fragment parts does not provide information on blind or non-blind fragments, but provides information on the fragment length distribution of the real (i.e. biological) 4C-seq library. In contrast to the regular virtual fragment library for 4C-seq data, fragments between two adjacent secondary restriction enzyme sites are counted as well.

Value

Data frame with lengths and corresponding frequences of fragments

Note

- The resulting fragment lengths and corresponding frequencies can easily be visualized with R's plot function or the small convenience function `drawDigestionFragmentHistogram`
- The resulting table of fragment frequencies can easily be exported with R's `write.table` function

Author(s)

Carolin Walter

Examples

```
shortTestGenome = "ATCCATGTAGGCTAAGTACACATGTTAAGGTACAGTACAATTGCACGATCAT"
fragments = simulateDigestion("catg", "gtac", shortTestGenome)
head(fragments)
```

`visualizeViewpoint` *Draw a near-cis coverage plot for 4C-seq data*

Description

This method creates a plot of near-cis 4C-seq fragment data around the experiment's viewpoint. Fragment-based raw data is visualized as grey dots, interpolated data (running median / running mean) as coloured dots. Trend line and quantiles are loess-smoothed; the trend line is shown as colored line whereas the quantiles are depicted as light-grey bands. A corresponding quantile legend is added in an extra plot.

Usage

```
visualizeViewpoint(expData, poi = data.frame(chr = character(), start = character(), end = character(),
```

Arguments

expData	Experiment data of class Data4Cseq with information on the 4C-seq experiment, including normalized near-cis fragment data for visualization
poi	Points of interest that will be marked in the plot
plotFileName	Name for the 4C-seq plot file
windowLength	Length of the window for running median / running mean that is used to smooth the trend line
interpolationType	Type of interpolation, either running median or running mean
picDim	Dimensions of the plot
maxY	Maximum y-value to plot. If no maximum is given, the maximum running median / mean value is used
minQuantile	Minimum quantile to draw
maxQuantile	Maximum quantile to draw
mainColour	Main colour of the plot
plotTitle	Title of the 4C-seq plot, depicted above the main plot
loessSpan	Span value for the loess curve; smaller values mean a tighter fit to the data points, but a value that is too small may produce errors
xAxisIntervalLength	Length of the x axis intervals in the plot
yAxisIntervalLength	Length of the y axis intervals in the plot
useFragEnds	Indicates whether fragment end data is used directly or interpolated on fragment level

Value

A near-cis coverage plot and a corresponding quantile legend

Note

PDF export and output as TIFF format are supported. The export format is chosen depending on the plot file name's ending. If no plot file name is provided, the result is plotted on screen.

Author(s)

Carolin Walter

Examples

```
data(liverData)
file <- system.file("extdata", "fetalLiverVP.bed", package="Basic4Cseq")
 visualizeViewpoint(liverData, readPointsOfInterestFile(file), plotFileName = "", mainColour = "red", plotTit
```

Index

∗ **Data4Cseq**
Data4Cseq, 6
∗ **checkRestrictionEnzymeSequence**
checkRestrictionEnzymeSequence, 2
∗ **chooseNearCisFragments**
chooseNearCisFragments, 3
∗ **classes**
Data4Cseq-class, 7
∗ **createVirtualFragmentLibrary**
createVirtualFragmentLibrary, 4
∗ **datasets**
liverData, 14
liverDataRaw, 15
∗ **drawDigestionFragmentHistogram**
drawDigestionFragmentHistogram, 9
∗ **drawHeatmap**
drawHeatmap, 10
∗ **exportVisualizationFragmentData**
exportVisualizationFragmentData, 11
∗ **getReadDistribution**
getReadDistribution, 12
∗ **giveEnzymeSequence**
giveEnzymeSequence, 13
∗ **importVisualizationFragmentData**
importVisualizationFragmentData, 14
∗ **normalizeFragmentData**
normalizeFragmentData, 16
∗ **plotTransInteractions**
plotTransInteractions, 16
∗ **prepare4CseqData**
prepare4CseqData, 18
∗ **printBEDFragmentLibrary**
printBEDFragmentLibrary, 19
∗ **printWigFile**
printWigFile, 20
∗ **readPointsOfInterestFile**
readPointsOfInterestFile, 21
∗ **readsToFragments**
readsToFragments, 22
∗ **simulateDigestion**
simulateDigestion, 22
∗ **visualizeViewpoint**
visualizeViewpoint, 23

checkRestrictionEnzymeSequence, 2, *19*
checkRestrictionEnzymeSequence,character,character-method *(checkRestrictionEnzymeSequence)*, 2
chooseNearCisFragments, 3
chooseNearCisFragments,Data4Cseq,numeric-method *(chooseNearCisFragments)*, 3
createVirtualFragmentLibrary, 4
createVirtualFragmentLibrary,BSgenome,character,character, *(createVirtualFragmentLibrary)*, 4
createVirtualFragmentLibrary,DNAString,character,character *(createVirtualFragmentLibrary)*, 4

Data4Cseq, 6
Data4Cseq,character,numeric,numeric,data.frame,GAlignments *(Data4Cseq)*, 6
Data4Cseq,character,numeric,numeric,missing,missing-method *(Data4Cseq)*, 6
Data4Cseq-class, 7
drawDigestionFragmentHistogram, 9
drawDigestionFragmentHistogram,data.frame-method *(drawDigestionFragmentHistogram)*, 9
drawHeatmap, 10
drawHeatmap,data.frame-method *(drawHeatmap)*, 10
drawHeatmap,Data4Cseq-method *(drawHeatmap)*, 10

exportVisualizationFragmentData, 11

exportVisualizationFragmentData,Data4Cseq,character-method
(exportVisualizationFragmentData), 11

getReadDistribution, 12
getReadDistribution,Data4Cseq-method
(getReadDistribution), 12
giveEnzymeSequence, 13
giveEnzymeSequence,character,character-method
(giveEnzymeSequence), 13

importVisualizationFragmentData, 14
importVisualizationFragmentData,character-method
(importVisualizationFragmentData), 14

liverData, 14
liverDataRaw, 15

nearCisFragments (Data4Cseq-class), 7
nearCisFragments,Data4Cseq-method
(Data4Cseq-class), 7
nearCisFragments<- (Data4Cseq-class), 7
nearCisFragments<-,Data4Cseq,data.frame-method
(Data4Cseq-class), 7
normalizeFragmentData, 16
normalizeFragmentData,Data4Cseq-method
(normalizeFragmentData), 16

plotTransInteractions, 16
plotTransInteractions,character,character,numeric,character-method
(plotTransInteractions), 16
plotTransInteractions,data.frame,character,numeric,data.frame-method
(plotTransInteractions), 16
pointsOfInterest (Data4Cseq-class), 7
pointsOfInterest,Data4Cseq-method
(Data4Cseq-class), 7
pointsOfInterest<- (Data4Cseq-class), 7
pointsOfInterest<-,Data4Cseq,data.frame-method
(Data4Cseq-class), 7
prepare4CseqData, 18
prepare4CseqData,character,character,character,character-method
(prepare4CseqData), 18
printBEDFragmentLibrary, 19
printBEDFragmentLibrary,character,character-method
(printBEDFragmentLibrary), 19
printWigFile, 20
printWigFile,Data4Cseq-method
(printWigFile), 20

rawFragments (Data4Cseq-class), 7
rawFragments,Data4Cseq-method
(Data4Cseq-class), 7
rawFragments<- (Data4Cseq-class), 7
rawFragments<-,Data4Cseq,data.frame-method
(Data4Cseq-class), 7
rawReads (Data4Cseq-class), 7
rawReads,Data4Cseq-method
(Data4Cseq-class), 7
rawReads<- (Data4Cseq-class), 7
rawReads<-,Data4Cseq,GAlignments-method
(Data4Cseq-class), 7
readLength (Data4Cseq-class), 7
readLength,Data4Cseq-method
(Data4Cseq-class), 7
readLength<- (Data4Cseq-class), 7
readLength<-,Data4Cseq,numeric-method
(Data4Cseq-class), 7
readPointsOfInterestFile, 21
readPointsOfInterestFile,character-method
(readPointsOfInterestFile), 21
readsToFragments, 22
readsToFragments,Data4Cseq,character-method
(readsToFragments), 22

simulateDigestion, 22
simulateDigestion,character,character,BSgenome-method
(simulateDigestion), 22
simulateDigestion,character,character,character-method
(simulateDigestion), 22
viewpointChromosome (Data4Cseq-class), 7
viewpointChromosome,Data4Cseq-method
(Data4Cseq-class), 7
viewpointChromosome<-
(Data4Cseq-class), 7
viewpointChromosome<-,Data4Cseq,character-method
(Data4Cseq-class), 7
viewpointInterval (Data4Cseq-class), 7
viewpointInterval,Data4Cseq-method
(Data4Cseq-class), 7
viewpointInterval<- (Data4Cseq-class), 7
viewpointInterval<-,Data4Cseq,numeric-method
(Data4Cseq-class), 7
visualizeViewpoint, 23
visualizeViewpoint,data.frame-method
(visualizeViewpoint), 23
visualizeViewpoint,Data4Cseq-method
(visualizeViewpoint), 23