Package ‘Scale4C’

March 7, 2024

Type Package
Title Scale4C: an R/Bioconductor package for scale-space transformation of 4C-seq data
Version 1.24.0
Date 2017-06-06
Author Carolin Walter
Maintainer Carolin Walter <carolin.walter@uni-muenster.de>
Imports methods, grDevices, graphics, utils
Depends R (>= 3.4), smoothie, GenomicRanges, IRanges, SummarizedExperiment

Description Scale4C is an R/Bioconductor package for scale-space transformation and visualization of 4C-seq data. The scale-space transformation is a multi-scale visualization technique to transform a 2D signal (e.g. 4C-seq reads on a genomic interval of choice) into a tesselation in the scale space (2D, genomic position x scale factor) by applying different smoothing kernels (Gauss, with increasing sigma). This transformation allows for explorative analysis and comparisons of the data's structure with other samples.

License LGPL-3

biocViews Visualization, QualityControl, DataImport, Sequencing, Coverage
git_url https://git.bioconductor.org/packages/Scale4C
git_branch RELEASE_3_18
git_last_commit 3e54c07
git_last_commit_date 2023-10-24
Repository Bioconductor 3.18
Date/Publication 2024-03-06
R topics documented:

addPointsOfInterest ........................................... 2
calculateFingerprintMap ..................................... 3
calculateScaleSpace ............................................ 4
findSingularities .............................................. 5
importBasic4CseqData ......................................... 6
liverData ......................................................... 7
liverDataVP ..................................................... 8
outputScaleSpaceTree ........................................... 8
plotInflectionPoints ........................................... 9
plotTessellation ................................................ 11
plotTraceback .................................................. 12
Scale4C ......................................................... 13
Scale4C-class ................................................... 14

Index 16

addPointsOfInterest Add points of interest to a Scale4C object

Description

The function addPointsOfInterest adds marker points to a Scale4C object, which are subsequently used to mark points of interest in created plots.

Usage

addPointsOfInterest(data, poi)

Arguments

data Scale4C object the points are to be added to

poi Points of interest data, in a GRanges object. Important: column names must be specified and include 'colour' and 'name' for each point of interest with appropriate values

Details

The function addPointsOfInterest adds predefined points of interest to a Scale4C object. Each point of interest is defined by 'chr', 'start', 'end', 'colour', and 'name'. A bed file or text file can be used to store the information, however, column names have to be added before import. Other additional columns are ignored by the function. The function then converts the information to a GRanges object.

Value

A data frame that contains the data for all points of interest
calculateFingerprintMap

**Author(s)**
Carolin Walter

**Examples**

```r
# import provided point of interest example, and check if import was
# successful
data(liverData)
poiFile <- system.file("extdata", "vp.txt", package="Scale4C")
pointsOfInterest(liverData) <- addPointsOfInterest(liverData,
  read.csv(poiFile, sep = "\t", stringsAsFactor = FALSE))
head(pointsOfInterest(liverData))
```

**calculateFingerprintMap**

*Calculate scale-space fingerprint map for given 4C-seq data*

**Description**

calculateFingerprintMap uses the scale space map to calculate the inflection points of the smoothed signals.

**Usage**

calculateFingerprintMap(data, maxSQSigma = 5000, epsilon = 0.0000001)

**Arguments**

data Scale-space object for the 4C-seq data
maxSQSigma Maximum square sigma used to calculate the fingerprint map
epsilon Small numeric value (can also be zero); used to test for inflection points

**Details**

Scale4C uses Gauss kernels of increasing (square) sigma to smooth the original 4C-seq signal. The resulting inflection points for a chosen sigma are stored in the corresponding line of the fingerprint map, i.e. a 2D matrix (position x sigma).

**Value**

A Scale4C object containing the fingerprint map for a Scale4C object, i.e. a matrix with quite a lot of zeros and the occasional 2 or -1 as symbols for inflection points. The fingerprint map is included as second assay of the Scale4C object’s scaleSpace SummarizedExperiment slot.

**Author(s)**

Carolin Walter
**Examples**

```r
# read prepared example data
data(liverData)
# use small maxSQSigma for a fast example
liverData<-calculateFingerprintMap(liverData, maxSQSigma = 50)
head(t(assay(scaleSpace(liverData), 2))[1:10,1:20])
```

---

**calculateScaleSpace**  
*Calculate scale-space representation for given 4C-seq data*

**Description**

Scale4C uses Gauss kernels of increasing (square) sigma to smooth the original 4C-seq signal. The resulting data is stored in a 2D matrix (position x sigma).

**Usage**

```r
calculateScaleSpace(data, maxSQSigma = 5000)
```

**Arguments**

- `data`  
  Scale-space object for the 4C-seq data
- `maxSQSigma`  
  Maximum square sigma used to calculate the scale space

**Details**

The central idea of the scale-space transformation is to smooth the original signal with increasing strength, identify inflection points, track those inflection points throughout the different smoothing layers, and find singularities in those inflection point 'lines'. In case of 4C-seq data, this corresponds to smoothing the signal gradually, while making notes when features such as 'peaks' or 'valleys' disappear by merging with other features. `calculateScaleSpace` smoothes the original signal up to a provided smoothing factor square sigma (Gauss kernel).

**Value**

A `SummarizedExperiment` that contains in its first assay the scale space representation for a `Scale4C` object

**Author(s)**

Carolin Walter

**Examples**

```r
# read prepared example data
data(liverData)
# use small maxSQSigma for a fast example
scaleSpace(liverData)<-calculateScaleSpace(liverData, maxSQSigma = 10)
head(t(assay(scaleSpace(liverData), 1))[1:10,1:20])
```
findSingularities

Find singularities in a provided scale-space fingerprint map

Description

This function allows to identify singular points in a scale-space fingerprint map.

Usage

findSingularities(data, minSQSigma = 5, outputTrackingInfo = FALSE, guessViewpoint = FALSE, useIndex = TRUE)

Arguments

data
minSQSigma
outputTrackingInfo
guessViewpoint
useIndex

 Scalar-space object for the 4C-seq data
Minimum square sigma used to calculate singularities; for a square sigma of 1, the data can be quite chaotic and identified singularities are less prone to error when a minSQSigma of 2 or higher is used
If TRUE, notify the user that a certain position / singularity causes problems during tracking in the fingerprint map
If TRUE, add another 'peak' at the coordinates of the viewpoint, if provided. Extra singularities can also be added manually. The idea is to decrease running speed significantly by not actually calculating the largest singularity for a typical 4C-seq experiment, i.e. the main viewpoint peak. Its inflection point contours should easily be visible in the fingerprint map, provided that the viewpoint position is actually included in the raw data, but calculating the full contours requires a very high sigma that should usually not be needed to identify other singularities in the area. Cave: Viewpoint contours don’t have to start directly next to the viewpoint coordinates.
If TRUE, use fragment index instead of genomic position data

Details

findSingularities identifies possible singular points in the fingerprint map's contours, i.e. points where a line of '2' and '-1' in the matrix meet. Starting from those points in scale-space, the contours are traced back down. This 'localization step' ensures that the coordinates for a feature ('peak' or 'valley') corresponding to a given singular point are as accurate as possible: Smoothing with a high-sigma Gauss kernel distorts the original signal somewhat, so that the inflection points identifying the start and the end of a certain feature 'move outwards'.

Value

A data frame that lists the position where a singular point occurs (genomic position and scale-space sigma), plus the size of the feature as given by its minimal / left and maximal / right position.
**importBasic4CseqData**

**Author(s)**

Carolin Walter

**Examples**

```r
# read prepared example data
data(liverData)
singularities(liverData) = findSingularities(liverData, 5, useIndex = TRUE)
singularities(liverData)
```

**Description**

A convenience function to easily include Basic4Cseq output data into Scale4C, `importBasic4CseqData` extracts valid fragments or valid fragment ends from a typical Basic4Cseq output table.

**Usage**

```r
importBasic4CseqData(rawFile, viewpoint, viewpointChromosome, distance, useFragEnds = TRUE)
```

**Arguments**

- **rawFile**: Name for the raw file
- **viewpoint**: Viewpoint position: only fragments around a certain point of interest are imported (doesn’t have to be the actual viewpoint of the experiment, though)
- **viewpointChromosome**: Viewpoint chromosome of the experiment
- **distance**: Distance from the viewpoint: only fragments within a certain distance of the viewpoint are imported
- **useFragEnds**: If TRUE, use full fragment end data, if FALSE, merge fragmentStart and fragmentEnd to a single item per fragment

**Details**

`importBasic4CseqData` is a convenience function to import data from Basic4Cseq. It can be ignored altogether if raw experimental data is imported from another source or with another function into R.

**Value**

A GRanges object that includes the experiment’s raw data for further processing
Author(s)
Carolin Walter

Examples

csvFile <- system.file("extdata", "liverData.csv", package="Scale4C")
liverReads <- importBasic4CseqData(csvFile, viewpoint = 21160072,
                                 viewpointChromosome = "chr10", distance = 1000000)
head(liverReads)

---

liverData

Example 4C-seq data set of fetal liver data

Description
This data set contains an instance of a Scale4C object.
The 4C-seq data was taken from Stadhouders et al’s fetal liver data set.

Usage
data("liverData")

Format
Formal class 'Scale4C'

Value
A pre-computed instance of a Scale4C object with fingerprint map and singularities. Scale-space image is reduced to save space.

Source
Shortened version of Stadhouders et al’s fetal liver data:

Examples
data("liverData")
liverData
liverDataVP

Example 4C-seq data set of fetal liver data, with added VP

Description
This data set contains an instance of a Scale4C object.
The 4C-seq data was taken from Stadhouders et al’s fetal liver data set. It contains a manually added viewpoint peak and a peak with a manual coordinate correction.

Usage
data("liverDataVP")

Format
Formal class 'Scale4C'

Value
A pre-computed instance of a Scale4C object with fingerprint map and singularities

Source
Shortened version of Stadhouders et al’s fetal liver data:

Examples
data("liverDataVP")
liverDataVP

outputScaleSpaceTree

Output list of all features for a given scale-space map

Description
This function provides a list of features for a given fingerprint map in scale-space, with position and range of sigma for which the feature in question exists

Usage
outputScaleSpaceTree(data, outputPeaks = TRUE, useLog = TRUE, useIndex = TRUE)
Arguments

- **data**: a Scale4C object with singularity data
- **outputPeaks**: If TRUE, output GRanges peak list only, if FALSE, also output valley data in a larger table
- **useLog**: If TRUE, use a log2 transformation on the square sigma values (fewer changes and fewer singularities for high sigma, in contrast to low sigma)
- **useIndex**: If TRUE, use fragment position

Details

Similar to plotTesselation, outputScaleSpaceTree analyzes a list of singular points and calculates corresponding features, i.e. 'peaks' and 'valleys'. Each singular point marks the disappearance (or occurrence, depending on the view) of a feature in scale space: With increasing square sigma as smoothing parameter for the Gauss kernel, smaller features are merged into larger features. In case of Gauss smoothing, one feature is always surrounded by two features of the opposite type, e.g. a 'peak' is surrounded by two 'valleys'. If a 'peak' is smoothed out, it is replaced by a new valley formed of the former peak's adjacent valleys. The singularity list contains only direct information on those 'central' features; outputScaleSpaceTree adds data on the direct neighbours/adjacent features and also provides the sigma ranges for the features as a measure of their stability throughout the smoothing process. Mean read counts for the identified features are also provided ("signal"). If outputPeaks is true, a reduced list of peaks is printed, while omitting valleys or the central-left-right structural information.

Value

A GRanges object that includes all features as identified through singular points, plus 'neighbour features' at each side (each 'peak' is surrounded by two 'valleys' and vice versa for Gauss kernel smoothing), with positions and range of sigma for which the feature in question remains stable

Author(s)

Carolin Walter

Examples

```r
# read prepared example data
data(liverDataVP)
output = outputScaleSpaceTree(liverDataVP, useLog = FALSE)
head(output)
```

plotInflectionPoints  
*Draw a smoothed near-cis profile with marked inflection points for a 4C-seq signal*
Description

plotInflectionPoints plots the inflection points for a given square sigma (i.e. a row of the fingerprint map) onto a corresponding smoothed near-cis plot for the 4C-seq signal. This allows to check problematic parts of the fingerprint map in more detail (e.g. unclear tracking areas with close contours), and to improve possible corrections in the singularity list. Plotting the smoothed signal for a given square sigma before calculation of the fingerprint map is also possible.

Usage

plotInflectionPoints(data, sqsigma, fileName = "inflectionPlot.pdf", width = 9, height = 5, maxVis = 5000, useIndex = TRUE, plotIP = TRUE)

Arguments

data
Scale4C object with experimental 4C-seq data to be smoothed and processed

sqsigma
Chosen square sigma, i.e. row of the fingerprint map to pick the inflection points from

fileName
Optional name for export file (pdf)

width
Width of the plot

height
Height of the plot

maxVis
Maximum y-axis value (read number, not sigma!) for visualization

useIndex
If TRUE, use fragment index for x-axis

plotIP
If TRUE, then mark chosen inflection points, if FALSE, simply plot smoothed data

Value

A near-cis plot of the smoothed data with (optional) marked inflection points in darker or lighter grey, depending on their direction

Note

PDF export is supported. If no plot file name is provided, the result is plotted on screen.

Author(s)

Carolin Walter

Examples

data(liverData)
plotInflectionPoints(liverData, 50)
plotTesselation

Draw the final scale space tesselation

Description

This method draws the final scale space tesselation, as specified by the list of singularities identified for a Scale4C object. Features are marked with different colours; for the default colour scheme, brown corresponds to 'peaks' and blue to 'valleys', while slightly darker colours mark features originating from singularities ('central' features in a set of three features, e.g. 'valley-peak-valley' or 'peak-valley-peak') and lighter colours the two adjacent features. Different colours for 'central' and 'adjacent' features allow for optical quality control of the tesselation: a 'central' / dark feature's direct predecessor or successor (y-axis) can't be of the same colour (i.e. a 'peak' that passes through a singularity is smoothed out into a 'valley'), and neighbouring intervals have to be of the opposing (but lighter) colour (i.e. each 'peak' is surrounded by two 'valleys' for Gauss kernel smoothing). The same is not necessarily true for an 'adjacent' / light feature, however.

Usage

plotTesselation(data, minSQSigma = 5, maxSQSigma = -1, maxVis = -1, fileName = "tesselationPlot.pdf", width = 5, height = 5, xInterval = 100, yInterval = 50, chosenColour = c("grey50", "moccasin", "lightskyblue1", "beige", "azure"), useIndex = TRUE)

Arguments

data Scale4C object with singularity data
minSQSigma Minimum square sigma to consider
maxSQSigma Maximum square sigma to consider; if -1 then the number of rows in the fingerprint map is used
maxVis Maximum y value for visualization (doesn't have to be maxSQSigma); if -1 also defaults to number of rows in the fingerprint map
fileName Optional name for export file (pdf)
width Width of the plot
height Height of the plot
xInterval Interval length for x-axis
yInterval Interval length for y-axis
chosenColour Chosen colours for the tesselation plot, five in total. Colour 1 is used for the actual lines of the plot, colour 2 for 'central peaks', colour 3 for 'central valleys', colour 4 for 'adjacent peaks', and colour 5 for 'adjacent valleys'
useIndex If TRUE, use fragment index for x-axis

Value

A tesselation plot, showing different features of the scale space with their range of existence (square sigma) and position
plotTraceback

Note
PDF export is supported. If no plot file name is provided, the result is plotted on screen.

Author(s)
Carolin Walter

Examples

```r
if(interactive()) {
  data(liverData)
  plotTesselation(liverData)
}
```

plotTraceback

Draw the traceback results for a list of singular points on a fingerprint map

Description
This method plots the traceback results together with fingerprint data, allowing to check for possible errors during tracking. Problems during tracking can occur if contours are very close, have holes, or if the singularity in question is not recognized at all due to holes at the meeting point of both contours that form a singular point. Each singular point is marked with a grey triangle, and the traced left and right end of the corresponding feature are connected with grey lines. If a contour’s end doesn’t match the traceback line, manual correction is possible in the singularity list.

Usage

```r
plotTraceback(data, maxSQSigma = -1, fileName = "tracebackPlot.pdf",
width = 15, height = 15, useIndex = TRUE)
```

Arguments

data Scale4C object with singularity data
maxSQSigma Maximum square sigma (i.e. maximum y value) to be drawn; if -1 then all available rows in the fingerprint map are used
fileName Optional name for export file (pdf)
width Width of the plot
height Height of the plot
useIndex If TRUE, use fragment index for x-axis

Value
A traceback plot, showing the traced singular points with their points of origin throughout different smoothing layers)
Note

PDF export is supported. If no plot file name is provided, the result is plotted on screen.

Author(s)

Carolin Walter

Examples

```r
if(interactive()) {
  data(liverData)
  plotTraceback(liverData)
}
```

---

**Scale4C**  
*Creating a Scale4C object*

**Description**

This function creates a Scale4C object. Data on the 4C-seq experiment, i.e. read counts per fragment and viewpoint coordinates, are stored and checked for plausibility.

**Usage**

```
Scale4C(viewpoint, viewpointChromosome, rawData)
```

**Arguments**

- `viewpoint` The experiment’s viewpoint (start, single coordinate)
- `viewpointChromosome` The experiment’s viewpoint Chromosome
- `rawData` Reads of the 4C-seq experiment per fragment on an interval of interest (GRanges object with position and read data)

**Details**

A Scale4C object contains the basic information on a 4C-seq experiment for a certain interval of interest, i.e. read counts at given positions. See **Scale4C-class** for details. Scale-space features such as fingerprint maps or tessellation are calculated during further steps of the analysis by the appropriate functions.

Scale4C expects the raw data to be in a simple data frame consisting of ‘position’ and ‘reads’. `importBasic4CseqData` allows to import fragment data from Basic4Cseq for convenience, however, preparing and importing a simple table with two columns into R is sufficient.

**Value**

An instance of the Scale4C class.
Author(s)

Carolin Walter

See Also

Scale4C-class

Examples

# create a Scale4C object from a Basic4Cseq export table with added
# viewpoint data
csvFile <- system.file("extdata", "liverData.csv", package="Scale4C")
liverReads <- importBasic4CseqData(csvFile, viewpoint = 21160072,
  viewpointChromosome = "chr10", distance = 1000000)
liverData = Scale4C(rawData = liverReads, viewpoint = 21160072,
  viewpointChromosome = "chr10")
liverData

Scale4C-class

Class "Scale4C"

Description

This class is a container for information on a specific 4C-seq scale-space transformation. Stored
information includes raw read data, the experiment’s viewpoint location (optional), possible points
of interest, the scale-space fingerprint map, and a list of identified singularities in scale-space.

Objects from the Class

Objects can be created by calls of the form new("Scale4C", ...).

Slots

viewpoint: Object of class "numeric" representing the viewpoint's location
viewpointChromosome: Object of class "character" representing the viewpoint's chromosome
pointsOfInterest: Object of class "GRanges" representing any points of interest to be marked
  in the visualizations (usually near-cis based, i.e. close to the viewpoint)
rawData: Object of class "GRanges" representing the 4C-seq reads (or signal strength) of the ex-
  periment at given genomic positions
scaleSpace: Object of class "SummarizedExperiment" representing the gradually smoothed 4C-
  seq signal ('scale space') in its first assay and the corresponding fingerprint map in its second
  assay.
singularities: Object of class "GRanges" representing singularities in the fingerprint map for
  the given 4C-seq signal
Methods

viewpoint<- signature(object = "Scale4C", value = "numeric"): Setter-method for the viewpoint slot.

viewpoint signature(object = "Scale4C"): Getter-method for the viewpoint slot.

viewpointChromosome<- signature(object = "Scale4C", value = "character"): Setter-method for the viewpointChromosome slot.

viewpointChromosome signature(object = "Scale4C"): Getter-method for the viewpointChromosome slot.

pointsOfInterest<- signature(object = "Scale4C", value = "GRanges"): Setter-method for the pointsOfInterest slot.

pointsOfInterest signature(object = "Scale4C"): Getter-method for the pointsOfInterest slot.

rawData<- signature(object = "Scale4C", value = "GRanges"): Setter-method for the rawData slot.

rawData signature(object = "Scale4C"): Getter-method for the rawData slot.

scaleSpace<- signature(object = "Scale4C", value = "matrix"): Setter-method for the scaleSpace slot.

scaleSpace signature(object = "Scale4C"): Getter-method for the scaleSpace slot.

singularities<- signature(object = "Scale4C", value = "GRanges"): Setter-method for the singularities slot.

singularities signature(object = "Scale4C"): Getter-method for the singularities slot.

Author(s)

Carolin Walter

Examples

showClass("Scale4C")
Index

* **Scale4C**
  * Scale4C, 13
* **addPointsOfInterest**
  * addPointsOfInterest, 2
* **calculateFingerprintMap**
  * calculateFingerprintMap, 3
* **calculateScaleSpace**
  * calculateScaleSpace, 4
* **classes**
  * Scale4C-class, 14
* **datasets**
  * liverData, 7
  * liverDataVP, 8
* **findSingularities**
  * findSingularities, 5
* **importBasic4CseqData**
  * importBasic4CseqData, 6
* **outputScaleSpaceTree**
  * outputScaleSpaceTree, 8
* **plotInflectionPoints**
  * plotInflectionPoints, 9
* **plotTesselation**
  * plotTesselation, 11
* **plotTraceback**
  * plotTraceback, 12

addPointsOfInterest, 2
addPointsOfInterest, Scale4C, data.frame-method (addPointsOfInterest), 2
calculateFingerprintMap, 3
calculateFingerprintMap, Scale4C-method (calculateFingerprintMap), 3
calculateScaleSpace, 4
calculateScaleSpace, Scale4C-method (calculateScaleSpace), 4
findSingularities, 5
findSingularities, Scale4C-method (findSingularities), 5
importBasic4CseqData, 6
importBasic4CseqData, character, numeric, character, numeric-method (importBasic4CseqData), 6
liverData, 7
liverDataVP, 8
outputScaleSpaceTree, 8
outputScaleSpaceTree, Scale4C-method (outputScaleSpaceTree), 8
plotInflectionPoints, 9
plotInflectionPoints, Scale4C, numeric-method (plotInflectionPoints), 9
plotTesselation, 11
plotTesselation, Scale4C-method (plotTesselation), 11
plotTraceback, 12
plotTraceback, Scale4C-method (plotTraceback), 12
pointsOfInterest (Scale4C-class), 14
pointsOfInterest, Scale4C-method (Scale4C-class), 14
pointsOfInterest<-, Scale4C-class, 14
pointsOfInterest<-, Scale4C, GRanges-method (Scale4C-class), 14
rawData (Scale4C-class), 14
rawData<-, Scale4C-class, 14
rawData<-, Scale4C, GRanges-method (Scale4C-class), 14
Scale4C, 13
Scale4C-class, 14
scaleSpace (Scale4C-class), 14
scaleSpace, Scale4C-method (Scale4C-class), 14
INDEX

scaleSpace<-(Scale4C-class), 14
scaleSpace<-,Scale4C,SummarizedExperiment-method
  (Scale4C-class), 14
singularities (Scale4C-class), 14
singularities,Scale4C-method
  (Scale4C-class), 14
singularities<-(Scale4C-class), 14
singularities<-,Scale4C,GRanges-method
  (Scale4C-class), 14

viewpoint (Scale4C-class), 14
viewpoint,Scale4C-method
  (Scale4C-class), 14
viewpoint<-(Scale4C-class), 14
viewpoint<-,Scale4C,numeric-method
  (Scale4C-class), 14
viewpointChromosome (Scale4C-class), 14
viewpointChromosome,Scale4C-method
  (Scale4C-class), 14
viewpointChromosome<-(Scale4C-class), 14

viewpointChromosome<-,Scale4C,character-method
  (Scale4C-class), 14