Package: mariner (via r-universe)

June 5, 2026

Type Package

Title Mariner: Explore the Hi-Cs

Version 1.12.0

Description Tools for manipulating paired ranges and working with Hi-C data in R. Functionality includes manipulating/merging paired regions, generating paired ranges, extracting/aggregating interactions from `.hic` files, and visualizing the results. Designed for compatibility with plotgardener for visualization.

RoxygenNote 7.3.3

Depends R (>= 4.2.0)

Suggests knitr, testthat (>= 3.0.0), rmarkdown, ExperimentHub, marinerData, TxDb.Hsapiens.UCSC.hg38.knownGene, fields

Imports methods, S4Vectors, BiocGenerics, BiocManager, GenomicRanges, InteractionSet, data.table, stats, rlang, glue, assertthat, dplyr, magrittr, dbscan, purrr, progress, GenomeInfoDb, strawr (>= 0.0.91), DelayedArray, HDF5Array, abind, BiocParallel, IRanges, SummarizedExperiment, rhdf5, plotgardener, RColorBrewer, colourvalues, utils, grDevices, graphics, grid

biocViews FunctionalGenomics, Visualization, HiC

VignetteBuilder knitr

License MIT + file LICENSE

Encoding UTF-8

Config/testthat/edition 3

Collate 'AllClasses.R' 'AllGenerics.R' 'mariner.R' 'methods-CountMatrix.R' 'methods-GInteractions.R' 'utils.R' 'methods-InteractionArray.R' 'methods-InteractionJaggedArray.R' 'methods-InteractionMatrix.R' 'methods-JaggedArray.R' 'methods-MatrixSelection.R' 'methods-MergedGInteractions.R' 'methods-adjustEnrichment.R' 'methods-aggHicMatrices.R' 'methods-as_ginteractions.R' 'methods-assignToBins.R' 'methods-binRanges.R' 'methods-calcLoopEnrichment.R' 'methods-changePixelRes.R' 'methods-hdf5BlockApply.R'

'methods-makeRandomGInteractions.R' 'methods-mergePairs.R'
'methods-pileupBoundaries.R' 'methods-pileupDomains.R'
'methods-pileupPixels.R' 'methods-pixelsToMatrices.R'
'methods-plotBullseye.R' 'methods-plotMatrix.R'
'methods-pullHic.R' 'methods-regularize.R'
'methods-removeShortPairs.R' 'methods-shiftRanges.R'
'methods-snapToBins.R' 'zzz.R'

URL https://ericscottdavis.com/mariner/,
https://github.com/EricSDavis/mariner

Config/pak/sysreqs
cmake make libbz2-dev liblzma-dev libuv1-dev libxml2-dev libssl-dev xz-utils zlib1g-dev

Repository https://bioc-release.r-universe.dev

Date/Publication 2026-04-28 13:00:45 UTC

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

RemoteRef RELEASE_3_23

RemoteSha 212aa16d75088f8d81cd10a564c91cef524845c6

Contents

`mariner-package` *Mariner: Explore the Hi-Cs*

Description

'mariner' is an R/Bioconductor package for exploring Hi-C data. It enables users to flexibly manipulate, extract, and aggregate chromatin interaction data quickly and efficiently.

Details

Key Features:

- Manipulating Paired Ranges - Convert, bin, and shift paired genomic ranges.
- Clustering & Merging Interactions - Group nearby interactions and select one as representative.
- Extracting & Aggregating Interactions - Pull Hi-C pixels or matrices, then aggregate by files or interactions.
- Calculating Loop Enrichment - Determine loop enrichment to local background with selection functions to flexibility select foreground and background.

For more details on the features of 'mariner', read the vignette: 'browseVignettes(package="mariner")'

Author(s)

Maintainer: Eric Davis <ericscottdavis@outlook.com> (ORCID)

Authors:

- Sarah Parker <smp3800@gmail.com> (ORCID)

See Also

Useful links:

- https://ericscottdavis.com/mariner/
- https://github.com/EricSDavis/mariner

`aggHicMatrices` *Aggregate count matrices from InteractionArray objects*

Description

Aggregation of count matrices is done blocks to avoid large memory usage. Use ‘nBlocks‘ to control the number of blocks read into memory at once. Blocks are defined as ‘length(interactions(x))/nBlocks‘.

Usage

```
aggHicMatrices(
  x,
  by = NULL,
  FUN = sum,
  nBlocks = 5,
  verbose = TRUE,
  BPPARAM = bpparam(),
  compressionLevel = 0
)

## S4 method for signature 'InteractionArray'
aggHicMatrices(
  x,
  by = NULL,
  FUN = sum,
  nBlocks = 5,
  verbose = TRUE,
  BPPARAM = bpparam(),
  compressionLevel = 0
)
```

Arguments

`x`	InteractionArray object.
`by`	String (length one character vector) describing whether to aggregate by interactions, files, or neither (i.e. NULL as default).
`FUN`	Function to use for aggregating.
`nBlocks`	Number of blocks for block-processing arrays. Default is 5. Increase this for large datasets. To read and process all data at once, set this value to 1.

verbose Boolean (TRUE or FALSE) describing whether to report block-processing progress.

BPPARAM Parallelization params (passed to 'BiocParallel::bplapply()'). Default is the result of 'BiocParallel::bpparams()'. Parallel processing is not available when 'by=interactions'.

compressionLevel
Number (length one numeric vector) between 0 (Default) and 9 indicating the compression level used on HDF5 file.

Details

Since interactions are typically the largest dimension in an InteractionArray, using 'by=interactions' creates an HDF5-backed array to store these large arrays. Currently parallel processing for HDF5-backed arrays are not supported regardless of the value of 'BPPARAM'.

Both 'by=NULL' and 'by=files' support parallel processing.

Value

An aggregated 'DelayedArray' object. If 'by=interactions' or 'by=files' then a 3-dimensional 'DelayedArray' is returned. If 'by=NULL' (default) then A 2-dimensional 'DelayedMatrix' is returned.

Examples

```
## Load marinerData
if (!require("marinerData", quietly = TRUE))
    BiocManager::install("marinerData")

## Read .hic file paths
hicFiles <- c(
    marinerData::LEUK_HEK_PJA27_inter_30.hic(),
    marinerData::LEUK_HEK_PJA30_inter_30.hic()
)
names(hicFiles) <- c("FS", "WT")

## Read in loops as GInteractions object
loops <-
    WT_5kbLoops.txt() |>
    setNames("WT") |>
    read.table(header=TRUE) |>
    as_ginteractions(keep.extra.columns=FALSE)

## Removes the "chr" prefix for compatibility
## with the preprocessed hic files
GenomeInfoDb::seqlevelsStyle(loops) <- 'ENSEMBL'

## Expand pixel ranges with a 5 pixel buffer on either side
loops <-
    assignToBins(loops, binSize=100e3) |>
    pixelsToMatrices(buffer=5)

## Extract 10, 11x11 count matrices from 2 hic files
iarr <-
```

```
    loops[1:10] |>
    pullHicMatrices(binSize=100e3,
                    files=hicFiles)

## Aggregate all, by files, or by interactions
aggHicMatrices(x=iarr)
aggHicMatrices(x=iarr, by="files")
aggHicMatrices(x=iarr, by="interactions")
```

---

aggMetadata *Aggregate the metadata columns of merged pairs*

Description

Aggregate the metadata columns of merged pairs

Usage

```
aggMetadata(x, columns, funs)

## S4 method for signature
## 'MergedGInteractions,character,character_OR_function_OR_list'
aggMetadata(x, columns, funs)
```

Arguments

x	MergedGInteractions object.
columns	Character vector of columns to aggregate.
funs	Character vector of functions to apply to 'columns'.

Value

'x' with aggregated metadata columns

Examples

```
## Load marinerData
if (!require("marinerData", quietly = TRUE))
    BiocManager::install("marinerData")

bedpeFiles <- c(
    marinerData::FS_5kbLoops.txt(),
    marinerData::WT_5kbLoops.txt()
)
names(bedpeFiles) <- c("FS", "WT")

## Read in bedpeFiles as a list of GInteractions
## Use only first 1000 rows for fast example
```

```
giList <-
    lapply(bedpeFiles, read.table, header=TRUE, nrows=1000) |>
    lapply(as_ginteractions) |>
    setNames(gsub("^.*extdata/(.{2}).*$", "\\1", bedpeFiles))

## Add names describing the source and loop
giList <- lapply(seq_along(giList), \(i) {
    x <- giList[[i]]
    x$name <- paste0(names(giList)[i], "_loop_", length(x))
    return(x)
})

## Cluster & merge pairs
x <- mergePairs(x = giList,
                radius = 5e03)

## List loop names
aggMetadata(x, columns = "name", fun = "list")

## Aggregate values
aggMetadata(x, columns = c("APScoreAvg"), fun = "mean")
aggMetadata(x, columns = c("APScoreAvg", "avg"), fun = "mean")
aggMetadata(x, columns = c("APScoreAvg"), fun = c("mean", "median"))

## Custom functions
aggMetadata(x, columns = c("APScoreAvg"), fun = \(x) {
    ifelse(is.na(sd(x)), 0, sd(x))
})
```

`as_ginteractions` *Convert DataFrames to GInteraction objects*

Description

'as_ginteractions' takes a paired-interaction (i.e. BEDPE) formatted data-frame-like object and converts it to a GInteractions object. For convenience, 'makeGInteractionsFromDataFrame' can be used as an alias.

Usage

```
as_ginteractions(
  df,
  keep.extra.columns = TRUE,
  starts.in.df.are.0based = FALSE,
  ...
)

makeGInteractionsFromDataFrame(
```

```
  df,
  keep.extra.columns = TRUE,
  starts.in.df.are.0based = FALSE,
  ...
)

## S4 method for signature
## 'DF_OR_df_OR_dt,logical_OR_missing,logical_OR_missing'
makeGInteractionsFromDataFrame(df, keep.extra.columns, starts.in.df.are.0based)

## S4 method for signature
## 'DF_OR_df_OR_dt,logical_OR_missing,logical_OR_missing'
as_ginteractions(df, keep.extra.columns, starts.in.df.are.0based)
```

Arguments

df	A data.table, data.frame, or DataFrame object. Assumes that the first 6 columnns are in the format chr1, start1, end1 and chr2, start2, end2, representing each pair of interactions.
keep.extra.columns	TRUE or FALSE (the default). If TRUE, the columns in df that are not used to form the genomic ranges of the returned GRanges object are then returned as metadata columns on the object. Otherwise, they are ignored. If df has a width column, then it's always ignored.
starts.in.df.are.0based	TRUE or FALSE (the default). If TRUE, then the start positions of the genomic ranges in df are considered to be 0-based and are converted to 1-based in the returned GRanges object. This feature is intended to make it more convenient to handle input that contains data obtained from resources using the "0-based start" convention. A notorious example of such resource is the UCSC Table Browser (http://genome.ucsc.edu/cgi-bin/hgTables).
...	Additional arguments.

Value

GInteraction object

Examples

```
## data.frame
df <- data.frame(chr1 = "chr1", x1 = 10000, x2 = 20000,
                 chr2 = "chr1", y1 = 30000, y2 = 40000)
makeGInteractionsFromDataFrame(df)

## data.frame
df <- data.frame(chr1 = "chr1", x1 = 10000, x2 = 20000,
                 chr2 = "chr1", y1 = 30000, y2 = 40000)
as_ginteractions(df)

## data.table
```

```
library(data.table)
df <- data.table::data.table(chr1 = "chr1", x1 = 10000, x2 = 20000,
                             chr2 = "chr1", y1 = 30000, y2 = 40000)
as_ginteractions(df)

## DataFrame
library(S4Vectors)
df <- DataFrame(chr1 = "chr1", x1 = 10000, x2 = 20000,
                chr2 = "chr1", y1 = 30000, y2 = 40000)
as_ginteractions(df)

## Alias
df <- data.frame(chr1 = "chr1", x1 = 10000, x2 = 20000,
                 chr2 = "chr1", y1 = 30000, y2 = 40000,
                 pval = 0.05, dist = 10000)
makeGInteractionsFromDataFrame(df)

## Additional metadata
df <- data.frame(chr1 = "chr1", x1 = 10000, x2 = 20000,
                 chr2 = "chr1", y1 = 30000, y2 = 40000,
                 pval = 0.05, dist = 10000)
as_ginteractions(df)

## Remove additional metadata
as_ginteractions(df, keep.extra.columns = FALSE)

## Add 1 to starts (for 0-based programs)
as_ginteractions(df, starts.in.df.are.0based = TRUE)
```

`assignToBins` *Flexibly bin paired ranges*

Description

Paired range objects (like ‘GInteractions‘ or BEDPE-formatted ‘data.frame‘-like objects) can be binned separately for each set of ranges.

Usage

```
assignToBins(x, binSize, pos1 = "center", pos2 = "center", ...)

## S4 method for signature
## 'DF_OR_df_OR_dt,
##   numeric,
##   character_OR_numeric_OR_missing,
##   character_OR_numeric_OR_missing'
assignToBins(x, binSize, pos1, pos2)
```

```
## S4 method for signature
## 'GInteractions,
##   numeric,
##   character_OR_numeric_OR_missing,
##   character_OR_numeric_OR_missing'
assignToBins(x, binSize, pos1, pos2)
```

Arguments

x	'GInteractions' or 'data.frame'-like object with paired interactions.
binSize	Integer (numeric) vector describing the new size of each pair of ranges. Accepts up to 2 values for adjusting each pair.
pos1, pos2	Position within anchors to resize the bin. Can be a character or integer vector of length 1 or 'length(x)' designating the position for each element in 'x'. Character options are "start", "end" and "center". Integers are referenced from the start position for '+' and '*' strands and from the end position for the '-' strand.
...	Additional arguments.

Value

GInteractions-like object binned to 'binSize' by 'pos1' and 'pos2'.

Examples

```
## Construct interactions as data.frame
df1 <-
    data.frame(chr1 = "chr1", x1 = 10000, x2 = 20000,
               chr2 = "chr1", y1 = 30000, y2 = 40000)

## Assign each range to 20-kb bins from the start positions
assignToBins(x = df1,
         binSize = 20000,
         pos1 = 'start',
         pos2 = 'start')

## Construct GInteractions
library(InteractionSet)
gi1 <-
    data.frame(chr1 = "chr1", x1 = 10000, x2 = 20000,
               chr2 = "chr1", y1 = 30000, y2 = 40000) |>
    as_ginteractions()

## Assign each range to 20-kb bins from the start positions
assignToBins(x = gi1,
         binSize = 20000,
         pos1 = 'start',
         pos2 = 'start')
```

binRanges *Flexibly bin ranges*

Description

Flexibly bin ranges

Usage

```
binRanges(x, binSize, pos = "center")

## S4 method for signature 'GRanges,numeric,character_OR_numeric_OR_missing'
binRanges(x, binSize, pos = "center")
```

Arguments

x	'GRanges' object
binSize	Integer (numeric) describing the new size of each range.
pos	Position within range to resize the bin. Can be a character or integer vector of length 1 or 'length(x)' designating the position for each element in 'x'. Character options are "start", "end" and "center". Integers are referenced from the start position for '+' and '*' strands and from the end position for the '-' strand.

Value

'GRanges' object that has been shifted by 'pos' and assigned to bins of 'binSize'.

Examples

```
library(GenomicRanges)

## Create example GRanges
gr1 <- GRanges(seqnames = "chr1",
               ranges = IRanges::IRanges(start = rep(5000,3),
                                         end = rep(6000,3)),
               strand = c('+', '-', '*'))

gr2 <- gr1 |> promoters(upstream = 2000, downstream = 200)

## Binning the results
binRanges(x = gr1, binSize = 1000, pos = 'start')
binRanges(x = gr1, binSize = 1000, pos = 'end')
binRanges(x = gr1, binSize = 1000, pos = 'center')

## Bin after shifting back to TSS
binRanges(x = gr2, binSize = 1000, pos = 2000)
```

`calcLoopEnrichment` *Calculate loop enrichment over background.*

Description

Pulls Hi-C pixels and calculates the enrichment of the selected foreground ('fg') over the selected background ('bg').

Usage

```
calcLoopEnrichment(
  x,
  files,
  fg = selectCenterPixel(mhDist = 1, buffer = defaultBuffer()),
 bg = selectTopLeft(n = 4, buffer = defaultBuffer()) + selectBottomRight(n = 4, buffer =
    defaultBuffer()),
  FUN = function(fg, bg) median(fg + 1)/median(bg + 1),
  nBlocks = 5,
  verbose = TRUE,
  BPPARAM = bpparam(),
  ...
)

## S4 method for signature 'GInteractions,character'
calcLoopEnrichment(
  x,
  files,
  fg = selectCenterPixel(mhDist = 1, buffer = defaultBuffer()),
 bg = selectTopLeft(n = 4, buffer = defaultBuffer()) + selectBottomRight(n = 4, buffer =
    defaultBuffer()),
  FUN = function(fg, bg) median(fg + 1)/median(bg + 1),
  nBlocks = 5,
  verbose = TRUE,
  BPPARAM = bpparam(),
  ...
)

## S4 method for signature 'InteractionArray,missing'
calcLoopEnrichment(
  x,
  files,
  fg = selectCenterPixel(mhDist = 1, buffer = defaultBuffer()),
 bg = selectTopLeft(n = 4, buffer = defaultBuffer()) + selectBottomRight(n = 4, buffer =
    defaultBuffer()),
  FUN = function(fg, bg) median(fg + 1)/median(bg + 1),
  nBlocks = 5,
  verbose = TRUE,
```

```
  BPPARAM = bpparam(),
  ...
)
```

Arguments

x	GInteractions object or an InteractionArray object.
files	Character file paths to '.hic' files. Required only if GInteractions object is supplied for x.
fg	MatrixSelection object of matrix indices for the foreground.
bg	MatrixSelection object of matrix indices for the background.
FUN	Function with at least two parameters (i.e., 'fg', 'bg') defining how enrichment should be calculated. Must produce a single value (numeric of length one). The first and second parameters must represent fg and bg, respectively.
nBlocks	Number of blocks for block-processing arrays. Default is 5. Increase this for large datasets. To read and process all data at once, set this value to 1.
verbose	Boolean (TRUE or FALSE) describing whether to report block-processing progress.
BPPARAM	Parallelization params (passed to 'BiocParallel::bplapply()'). Default is the result of 'BiocParallel::bpparams()'. Parallel processing is not available when 'by=interactions'.
...	Additional arguments passed to 'pullHicMatrices'. See ?['pullHicMatrices'].

Value

A DelayedMatrix of enrichment scores where rows are interactions (i.e. loops) and columns are Hi-C files.

Examples

```
## Load marinerData
if (!require("marinerData", quietly = TRUE))
    BiocManager::install("marinerData")

## Read .hic file paths
hicFiles <- c(
    marinerData::LEUK_HEK_PJA27_inter_30.hic(),
    marinerData::LEUK_HEK_PJA30_inter_30.hic()
)
names(hicFiles) <- c("FS", "WT")

## Read in loops as GInteractions object
loops <-
    WT_5kbLoops.txt() |>
    setNames("WT") |>
    read.table(header=TRUE) |>
    as_ginteractions(keep.extra.columns=FALSE)

## Removes the "chr" prefix for compatibility
```

```
## with the preprocessed hic files
GenomeInfoDb::seqlevelsStyle(loops) <- 'ENSEMBL'

## Expand binSize of loops
loops <- assignToBins(x=loops, binSize=100e3)

## Calculate loop enrichment
calcLoopEnrichment(x=loops[1:10],
                   files=hicFiles)

## Customize different foreground/background
## with selection functions
buffer <- 10 # choose pixel radius around center
fg <- selectCenterPixel(mhDist=seq(0,4), buffer=buffer)
bg <- selectCorners(n=6, buffer=buffer) +
    selectOuter(n=2, buffer=buffer)

## Calculate loop enrichment
calcLoopEnrichment(x=loops[1:10],
                   files=hicFiles,
                   fg=fg,
                   bg=bg)

## Extract count matrices first
mats <- assignToBins(loops[1:10],100e3) |>
  pixelsToMatrices(buffer=10) |>
    pullHicMatrices(
    files=hicFiles,
    binSize=100e3)

## Calculate loop enrichment from count matrices
calcLoopEnrichment(x = mats)
```

`changePixelRes` *Change pixels from one resolution to another selecting the new pixel using Hi-C data.*

Description

A GInteractions object containing pixels of interest is resized to the ‘from‘ resolution (if its not already), then count matrices are extracted for each interaction and Hi-C file using the new ‘to‘ resolution. Count matrices are aggregated by interactions with the supplied ‘aggFUN‘, and a new pixel is selected with the supplied ‘selectFUN‘. For large datasets, increase ‘nBlocks‘ to allow for smaller blocks of data to be processed in memory.

Usage

```
changePixelRes(
  x,
```

```
  files,
  from,
  to,
  aggFUN = sum,
  selectFUN = "which.max",
  nBlocks = 5,
  verbose = TRUE,
  norm = "KR",
  half = "upper",
  ...
)

## S4 method for signature 'GInteractions,character'
changePixelRes(
  x,
  files,
  from,
  to,
  aggFUN = sum,
  selectFUN = "which.max",
  nBlocks = 5,
  verbose = TRUE,
  norm = "KR",
  half = "upper",
  ...
)
```

Arguments

x	GInteractions object.
files	Character file paths to ‘.hic‘ files.
from	Number (length one numeric vector) describing the resolution of ‘x‘. Data will be binned to this value if it is not already binned.
to	Number (length one numeric vector) describing the new resolution for the pixels.
aggFUN	Function to use for aggregating across Hi-C files. Must be passable to ‘which.max‘ or ‘which.min‘. Default is "sum".
selectFUN	Function to use for selecting among aggregated interactions. Must be one of "which.max" or "which.min".
nBlocks	Number of blocks for block-processing arrays. Default is 5. Increase this for large datasets. To read and process all data at once, set this value to 1.
verbose	Boolean (TRUE or FALSE) describing whether to report block-processing progress. Default is TRUE.
norm	String (length one character vector) describing the Hi-C normalization to apply. Use ‘strawr::readHicNormTypes()‘ to see accepted values for each file in ‘files‘.
half	String (character vector of length one) indicating whether to keep values for the upper triangular (‘half="upper"‘) where ‘start1 < start2‘, lower triangular

(‘half="lower"‘) where ‘start1 > start2‘, or both (‘half="both"‘, default). When ‘half="upper"‘ all lower triangular values are ‘NA‘. When ‘half="lower"‘ all upper triangular values are ‘NA‘. When ‘half="both"‘ there are no ‘NA‘ values. For interchromosomal interactions there is no inherent directionality between chromosomes, so data is returned regardless of specified order.

... Additional arguments passed to ‘pullHicMatrices()‘. See ?[‘pullHicMatrices‘].

Value

A GInteractions object with the updated pixel interactions, along with a column with the aggregated max/min value for that pixel.

Examples

```
## Load marinerData
if (!require("marinerData", quietly = TRUE))
    BiocManager::install("marinerData")

## Read .hic file paths
hicFiles <- c(
    marinerData::LEUK_HEK_PJA27_inter_30.hic(),
    marinerData::LEUK_HEK_PJA30_inter_30.hic()
)
names(hicFiles) <- c("FS", "WT")

## Read in loops as GInteractions object
loops <-
    WT_5kbLoops.txt() |>
    setNames("WT") |>
    read.table(header=TRUE) |>
    as_ginteractions(keep.extra.columns=FALSE)

## Removes the "chr" prefix for compatibility
## with the preprocessed hic files
GenomeInfoDb::seqlevelsStyle(loops) <- 'ENSEMBL'

## Rebin loops to 2.5e6 resolution
loops <- assignToBins(x=loops, binSize=2.5e06)

## Change pixel resolution from 2.5e6 to 500e3
changePixelRes(x=loops[1:5],
               files=hicFiles,
               from=2.5e6,
               to=500e3)
```

---

clusters *Get clustered pairs from MergedGInteractions object*

Description

Returns the clustered pairs associated with each range in the ‘MergedGInteractions‘ object. Order always follows the indices of the ‘MergedGInteractions‘ object.

Usage

```
clusters(x, ...)

## S4 method for signature 'MergedGInteractions'
clusters(x)
```

Arguments

x	MergedGInteractions object.
...	Additional arguments.

Value

A list of data.tables cooresponding to each pair in ‘x‘.

Examples

```
## Load required packages
library(data.table, include.only="fread")

## Load marinerData
if (!require("marinerData", quietly = TRUE))
    BiocManager::install("marinerData")

## Reference BEDPE files (loops called with SIP)
bedpeFiles <- c(
    marinerData::FS_5kbLoops.txt(),
    marinerData::WT_5kbLoops.txt()
)
names(bedpeFiles) <- c("FS", "WT")

## Read in bedpeFiles as a list of GInteractions
## Use only first 1000 rows for fast example
giList <-
    lapply(bedpeFiles, fread, nrows = 1000) |>
    lapply(as_ginteractions)

## Cluster & merge pairs
x <- mergePairs(x = giList,
                radius = 10e03,
                column = "APScoreAvg")

## Access pair clusters
clusters(x[1:3])
clusters(x[3:1])
clusters(x[c(3, 1, 2)])
```

```
clusters(x) |> length()
```

`counts,InteractionArray-method`
Access count matrices from InteractionArray or InteractionMatrix

---

Description

Access count matrices from InteractionArray or InteractionMatrix

Access count matrices from InteractionArray or InteractionMatrix

Replace method for counts

Usage

```
## S4 method for signature 'InteractionArray'
counts(object, showDimnames = FALSE)

## S4 method for signature 'InteractionMatrix'
counts(object)

## S4 replacement method for signature 'InteractionMatrix'
counts(object) <- value
```

Arguments

`object`	InteractionMatrix object
`showDimnames`	Logical vector of length-one indicating whether to show dimensions of count matrices (default FALSE). Only applicable for InteractionArray objects.
`value`	Value for replacement

Value

For InteractionArray, a 4-dimensional DelayedArray of Hi-C submatrices is returned with the following dimensions: rows of count matrix, columns of count matrix, Interactions in 'object', Hi-C 'files'.

For InteractionMatrix, a 2-dimensional DelayedArray is returned with rows representing interactions in 'object' and columns for each Hi-C file in 'files'.

For InteractionMatrix, the replace matrix replaces the counts assay with matrix-like objects supplied in 'value'.

Examples

```
## Load marinerData
if (!require("marinerData", quietly = TRUE))
    BiocManager::install("marinerData")

## Read .hic file paths
hicFiles <- c(
    marinerData::LEUK_HEK_PJA27_inter_30.hic(),
    marinerData::LEUK_HEK_PJA30_inter_30.hic()
)
names(hicFiles) <- c("FS", "WT")

#####################################
## Accessing Hi-C count submatrices ##
#####################################

## Create example interactions
x <- read.table(text="
        9 14435000 14490000 9 14740000 14795000
        9 89540000 89595000 9 89785000 89840000
        9 23700000 23755000 9 23760000 23815000")
x <- as_ginteractions(x)

## Extract 3, 11x11 count matrices from 2 hic files
iarr <- pullHicMatrices(x, hicFiles, 5e03)

## Access count matrices
counts(iarr)
counts(iarr, FALSE)

################################
## Accessing Hi-C count matrix ##
################################

## Create example interactions
x <- read.table(text="
        9 14000000 14500000 9 14500000 15000000
        9 89500000 90000000 9 89500000 90000000
        9 23500000 24000000 9 23500000 24000000")
x <- as_ginteractions(x)

## Extract 3 pixels from 2 hic files
imat <- pullHicPixels(x, hicFiles, 500e03)

## Access count matrix
counts(imat)

################################
## Replacing Hi-C count matrix ##
################################

## Realize as in-memory matrix
```

```
counts(imat) <- as.matrix(counts(imat))
counts(imat)
imat
```

`defaultBuffer` *Return default buffer If InteractionArray is supplied, it uses the dimensions of counts matrices to set the buffer dimensions.*

Description

Return default buffer If InteractionArray is supplied, it uses the dimensions of counts matrices to set the buffer dimensions.

Usage

```
defaultBuffer(x)
```

Arguments

x	InteractionArray

Value

5 (set default), the buffer of the provided InteractionArray, or an error message if the InteractionArray is not odd and square (no buffer)

`findOverlaps` *Overlap methods for InteractionJaggedArray*

Description

Overlap methods for InteractionJaggedArray

Usage

```
## S4 method for signature 'InteractionJaggedArray,InteractionJaggedArray'
findOverlaps(
  query,
  subject,
  maxgap = -1L,
  minoverlap = 0L,
  type = c("any", "start", "end", "within", "equal"),
  select = c("all", "first", "last", "arbitrary"),
  ignore.strand = TRUE,
  ...,
```

```
  use.region = "both"
)

## S4 method for signature 'InteractionJaggedArray,Vector'
findOverlaps(
  query,
  subject,
  maxgap = -1L,
  minoverlap = 0L,
  type = c("any", "start", "end", "within", "equal"),
  select = c("all", "first", "last", "arbitrary"),
  ignore.strand = TRUE,
  ...,
  use.region = "both"
)

## S4 method for signature 'InteractionJaggedArray,missing'
findOverlaps(
  query,
  subject,
  maxgap = -1L,
  minoverlap = 0L,
  type = c("any", "start", "end", "within", "equal"),
  select = c("all", "first", "last", "arbitrary"),
  ignore.strand = TRUE,
  ...,
  use.region = "both"
)

## S4 method for signature 'InteractionJaggedArray,InteractionJaggedArray'
countOverlaps(
  query,
  subject,
  maxgap = -1L,
  minoverlap = 0L,
  type = c("any", "start", "end", "within", "equal"),
  select = c("all", "first", "last", "arbitrary"),
  ignore.strand = TRUE,
  ...,
  use.region = "both"
)

## S4 method for signature 'InteractionJaggedArray,Vector'
countOverlaps(
  query,
  subject,
  maxgap = -1L,
  minoverlap = 0L,
```

```
  type = c("any", "start", "end", "within", "equal"),
  select = c("all", "first", "last", "arbitrary"),
  ignore.strand = TRUE,
  ...,
  use.region = "both"
)

## S4 method for signature 'InteractionJaggedArray,missing'
countOverlaps(
  query,
  subject,
  maxgap = -1L,
  minoverlap = 0L,
  type = c("any", "start", "end", "within", "equal"),
  select = c("all", "first", "last", "arbitrary"),
  ignore.strand = TRUE,
  ...,
  use.region = "both"
)

## S4 method for signature 'InteractionJaggedArray,InteractionJaggedArray'
overlapsAny(
  query,
  subject,
  maxgap = -1L,
  minoverlap = 0L,
  type = c("any", "start", "end", "within", "equal"),
  ...,
  use.region = "both"
)

## S4 method for signature 'InteractionJaggedArray,Vector'
overlapsAny(
  query,
  subject,
  maxgap = -1L,
  minoverlap = 0L,
  type = c("any", "start", "end", "within", "equal"),
  ...,
  use.region = "both"
)

## S4 method for signature 'InteractionJaggedArray,missing'
overlapsAny(
  query,
  subject,
  maxgap = -1L,
  minoverlap = 0L,
```

```
  type = c("any", "start", "end", "within", "equal"),
  ...,
  use.region = "both"
)

## S4 method for signature 'InteractionJaggedArray,InteractionJaggedArray'
subsetByOverlaps(
  x,
  ranges,
  maxgap = -1L,
  minoverlap = 0L,
  type = c("any", "start", "end", "within", "equal"),
  invert = FALSE,
  ...,
  use.region = "both"
)

## S4 method for signature 'InteractionJaggedArray,Vector'
subsetByOverlaps(
  x,
  ranges,
  maxgap = -1L,
  minoverlap = 0L,
  type = c("any", "start", "end", "within", "equal"),
  invert = FALSE,
  ...,
  use.region = "both"
)

## S4 method for signature 'InteractionJaggedArray,missing'
subsetByOverlaps(
  x,
  ranges,
  maxgap = -1L,
  minoverlap = 0L,
  type = c("any", "start", "end", "within", "equal"),
  invert = FALSE,
  ...,
  use.region = "both"
)
```

Arguments

`query`, `subject`, `x`, `ranges`
: An InteractionJaggedArray, Vector, GInteractions or InteractionSet object, depending on the specified method. At least one of these must be a ‘subject‘ can be missing if query is an InteractionJaggedArray object.

`maxgap`, `minoverlap`, `type`, `select`
: see ¿findOverlaps‘ in the GenomicRanges package.

ignore.strand	see ¿findOverlaps‘ in InteractionSet package for more information.
...	see ¿findOverlaps‘ in InteractionSet package for more information
use.region	see ¿findOverlaps‘ in InteractionSet package for more information.
invert	Boolean (TRUE/FALSE) to invert selection. Default is TRUE.

Value

‘findOverlaps‘ returns a Hits object. ‘countOverlaps‘ returns an integer vector of overlaps for each interaction in ‘query‘. ‘overlapsAny‘ returns a logical vector of overlaps for each interaction in ‘query‘. ‘subsetByOverlaps‘ returns overlapping interactions as an InteractionJaggedArray.

Examples

```
## Load marinerData
if (!require("marinerData", quietly = TRUE))
    BiocManager::install("marinerData")

## Read .hic file paths
hicFiles <- c(
    marinerData::LEUK_HEK_PJA27_inter_30.hic(),
    marinerData::LEUK_HEK_PJA30_inter_30.hic()
)
names(hicFiles) <- c("FS", "WT")

## Create test interactions
gi <- read.table(text="
            1 51000000 51300000 1 51000000 51500000
            2 52000000 52300000 3 52000000 52500000
            1 150000000 150500000 1 150000000 150300000
            2 52000000 52300000 2 52000000 52800000") |>
    as_ginteractions()

## InteractionJaggedArray object
iarr <- pullHicMatrices(gi, hicFiles, 100e03, half="both")

## Shift first two ranges out of range
gi2 <- c(assignToBins(gi[1:2], binSize=100e3, pos1=-200e3), gi[3:4])

## Find overlaps
findOverlaps(iarr, gi2)
countOverlaps(iarr, gi2)
countOverlaps(iarr, gi2, maxgap=100e3)
overlapsAny(iarr, gi2)
subsetByOverlaps(iarr, gi2)
subsetByOverlaps(iarr, gi2, invert=TRUE)
```

hdf5BlockApply *HDF5-backed blockApply*

Description

Read in array data in blocks, apply function, and write back to an HDF5 file.

Usage

```
hdf5BlockApply(x, FUN, sink, grid, sink_grid, verbose = TRUE)

## S4 method for signature 'DelayedArray'
hdf5BlockApply(x, FUN, sink, grid, sink_grid, verbose = TRUE)
```

Arguments

x	Delayed Array object.
FUN	Function that takes one argument 'block' and processes it.
sink	HDF5RealizationSink object.
grid	ArrayGrid over array 'x'.
sink_grid	ArrayGrid over 'sink'.
verbose	Logical - whether block processing progress should be displayed.

Details

Implements an HDF5-backed option for block processing on DelayedArray objects.

Value

An HDF5Array object.

Examples

```
## #################################################
## This function is intended for advanced users.
## To learn more about using DelayedArray
## or HDF5-backed objects, see ?DelayedArray or
## ?HDF5Array
#################################################

library(DelayedArray)
library(HDF5Array)
library(rhdf5)

## Create example array that is longer in the
## 3rd dimension (representing interactions)
dims <- c(11L, 11L, 100L, 2L)
```

```
a <- array(data=seq(1, prod(dims)), dim=dims)
a <- DelayedArray(a)

## Define spacings, breaking up the longest dim
## Here we are processing in blocks of 10
spacings <- dim(a)
spacings[3] <- ceiling(spacings[3]/10)

## Define storage dimensions (all except those
## over which the function is being applied)
storageDims <- dims[c(1,2,3)]

## Define chunk dimensions for writing to HDF5
chunkDims <- storageDims
chunkDims[3] <- spacings[3]

## Create grid for applying the data (grid)
## and grid for writing to the sink (sink_grid)
grid <- RegularArrayGrid(dims, spacings)
sink_grid <- RegularArrayGrid(storageDims, chunkDims)

## Create HDF5 file for writing
h5 <- tempfile(fileext = ".h5")
h5createFile(h5)

## Define compression for HDF5
compressionLevel <- 0

## Create HDF5-backed realization sink
sink <- HDF5RealizationSink(filepath=h5,
                            name="counts",
                            type="integer",
                            dim=storageDims,
                            chunkdim=chunkDims,
                            level=compressionLevel)

## Wrap function that operates on each block
## this can be anything, here it is sum
FUN <- \(block) apply(block, c(1,2,3), sum)

## Read, apply, and write to HDF5
ans <- hdf5BlockApply(x=a,
                      FUN=FUN,
                      sink=sink,
                      grid=grid,
                      sink_grid=sink_grid,
                      verbose=TRUE)
ans
```

`InteractionArray-class`
InteractionArray Class

Description

The 'InteractionArray' class extends 'InteractionSet' to provide an interface for accessing submatrices pulled from Hi-C data.

Usage

```
InteractionArray(assays, interactions, ...)

## S4 method for signature 'ANY,GInteractions'
InteractionArray(assays, interactions, ...)

## S4 method for signature 'missing,missing'
InteractionArray(assays, interactions, ...)

## S4 method for signature 'InteractionArray'
show(object)

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

## S4 method for signature 'InteractionArray'
cbind(..., deparse.level = 1)
```

Arguments

`assays, interactions`	See ?InteractionSet
`...`	InteractionArray objects to be combined column-wise. All objects must be the same class.
`object`	InteractionArray object.
`deparse.level`	An integer scalar; see '?base::cbind' for a description of this argument.

Details

This class is constructed with the 'pullHicMatrices()' function when all paired ranges have equal dimensions.

Value

An InteractionArray (see description)

See Also

[InteractionSet::InteractionSet]

Examples

```
InteractionArray()
```

`InteractionJaggedArray-class`
InteractionJaggedArray Class

Description

The 'InteractionJaggedArray' class creates a container for storing interaction data alongside irregular arrays. This allows the storage of matrices with different dimensions on-disk using HDF5.

Subset an InteractionJaggedArray by its interactions ([i,]) or its Hi-C files ([,j]).

Usage

```
## S4 method for signature 'InteractionJaggedArray'
show(object)

## S4 method for signature 'InteractionJaggedArray'
dim(x)

## S4 method for signature 'InteractionJaggedArray'
interactions(x)

## S4 method for signature 'InteractionJaggedArray'
metadata(x)

## S4 method for signature 'InteractionJaggedArray'
colData(x)

## S4 method for signature 'InteractionJaggedArray'
counts(object)

## S4 method for signature 'InteractionJaggedArray'
path(object)

## S4 method for signature 'InteractionJaggedArray'
length(x)

## S4 method for signature 'InteractionJaggedArray,ANY,ANY,ANY'
x[i, j]
```

Arguments

`object`	InteractionJaggedArray object.
`x`	An InteractionJaggedArray object.
`i`	Numeric vector indicating the indices of interactions to extract.
`j`	Numeric vector indicating the indices of files to extract.

Details

The object returned will be a InteractionJaggedArray if the submatrices contain different dimensions. However, the returned object will automatically be coerced into a InteractionArray if possible (i.e. the dimensions of the rows and columns of submatrices are the same.)

Value

'InteractionJaggedArray()' creates an InteractionJaggedArray object.

'dim()' returns a list of the dimensions of the interactions, files, and count matrices.

'interactions()' returns the interactions.

'metadata()' returns the metadata.

'colData()' returns the column data.

'counts()' returns the JaggedArray object containing count matrix information.

'path()' returns a character vector with the path to the HDF5 file with the JaggedArray data.

'length()' returns an integer with the number of interactions in an InteractionJaggedArray object.

Subsetting returns an InteractionJaggedArray or InteractionArray object (see Details).

Slots

`interactions` A GInteractions object.

`colData` Column data describing Hi-C files.

`counts` A JaggedArray object with data.

`metadata` List of metadata describing the object.

Examples

```
## Load marinerData
if (!require("marinerData", quietly = TRUE))
    BiocManager::install("marinerData")

## Read .hic file paths
hicFiles <- c(
    marinerData::LEUK_HEK_PJA27_inter_30.hic(),
    marinerData::LEUK_HEK_PJA30_inter_30.hic()
)
names(hicFiles) <- c("FS", "WT")

## Create test interactions
```

```
gi <- read.table(text="
            1 51000000 51300000 1 51000000 51500000
            2 52000000 52300000 3 52000000 52500000
            1 150000000 150500000 1 150000000 150300000
            2 52000000 52300000 2 52000000 52800000") |>
    as_ginteractions()

## InteractionJaggedArray object
iarr <- pullHicMatrices(gi, hicFiles, 100e03, half="both")
iarr

## Show dimensions
dim(iarr)

## Access interactions
interactions(iarr)

## Access metadata
metadata(iarr)

## Access colData
colData(iarr)

## Access count matrices
counts(iarr)

## Access path to HDF5 data
path(iarr)

## length
length(iarr)

## Subsetting
iarr[1:3,1]
```

`InteractionMatrix-class` *InteractionMatrix Class*

Description

The 'InteractionMatrix' class extends the 'InteractionSet' to provide an interface for accessing the count matrix pulled from Hi-C data.

Usage

```
InteractionMatrix(assays, interactions, ...)

## S4 method for signature 'ANY,GInteractions'
```

```
InteractionMatrix(assays, interactions, ...)

## S4 method for signature 'missing,missing'
InteractionMatrix(assays, interactions, ...)

## S4 method for signature 'InteractionMatrix'
show(object)

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

## S4 method for signature 'InteractionMatrix'
cbind(..., deparse.level = 1)
```

Arguments

assays, interactions	See ?InteractionSet
...	InteractionMatrix objects to be combined column-wise. All objects must be the same class.
object	InteractionMatrix object.
deparse.level	An integer scalar; see '?base::cbind' for a description of this argument.

Details

This class is constructed with the 'pullHicPixels()' function when all paired ranges define a single pixel.

Value

An InteractionMatrix (see description)

See Also

[InteractionSet::InteractionSet]

Examples

```
InteractionMatrix()
```

JaggedArray-class *JaggedArray Class*

Description

The 'JaggedArray' class creates a container for storing irregular or jagged array data. This allows the storage of matrices with different dimensions on-disk using HDF5.

Subset a JaggedArray by its interactions ([i,]) or its Hi-C files ([,j]).

'as.list' reads the on-disk data and returns it as an in-memory list of matrices.

Usage

```
## S4 method for signature 'JaggedArray'
show(object)

## S4 method for signature 'JaggedArray,ANY,ANY,ANY'
x[i, j, ..., drop = TRUE]

## S4 method for signature 'JaggedArray'
as.list(x)

## S4 method for signature 'JaggedArray'
path(object)

## S4 method for signature 'JaggedArray'
dim(x)
```

Arguments

object	JaggedArray object.
x	JaggedArray object.
i	Numeric vector indicating the indices of interactions to extract.
j	Numeric vector indicating the indices of files to extract.
...	Additional indices for subsetting multidimensional arrays.
drop	Not accepted for JaggedArray objects.

Details

NOTE: This class is designed specifically for holding a 4-dimensional JaggedArray <n x m x i x j> where n x m are rows and cols of count matrices, i is interactions, and j is Hi-C files.

The object returned will be a JaggedArray if the submatrices contain different dimensions. However, the returned object will automatically be coerced into a DelayedArray if possible (i.e. the dimensions of the rows and columns are the same.)

The JaggedArray data is still stored on-disk in an HDF5 file until it is coerced into a DelayedArray or realized as a list of matrices.

Value

'JaggedArray()' creates a JaggedArray object.

Subsetting returns a JaggedArray or DelayedArray object (see Details).

'as.list()' returns a list of matrices.

'path()' returns a character vector with the path to the HDF5 file with the JaggedArray data.

'dim()' returns a list of dimensions of the JaggedArray of rows, cols, interactions and files.

Slots

`h5File` path to file for creating and storing data as an HDF5 file.

`dim` dimensions describing the number of matrices contained. dim[1] is the number of interactions, dim[2] is the number of files.

`subList` is a list of length 2 where the first position refers to interactions and the second refers to files. This list is used to record subsetting operations which are then later applied when accessing data stored in the HDF5 file.

Examples

```
## Load marinerData
if (!require("marinerData", quietly = TRUE))
    BiocManager::install("marinerData")

## Read .hic file paths
hicFiles <- c(
    marinerData::LEUK_HEK_PJA27_inter_30.hic(),
    marinerData::LEUK_HEK_PJA30_inter_30.hic()
)
names(hicFiles) <- c("FS", "WT")

## Create test interactions
gi <- read.table(text="
            1 51000000 51300000 1 51000000 51500000
            2 52000000 52300000 3 52000000 52500000
            1 150000000 150500000 1 150000000 150300000
            2 52000000 52300000 2 52000000 52800000") |>
    as_ginteractions()

## InteractionJaggedArray object
iarr <- pullHicMatrices(gi, hicFiles, 100e03, half="both")
arr <- counts(iarr)
arr

## Subsetting
arr[,,1,] # DelayedArray
arr[,,,1] # JaggedArray

## Realize as list
as.list(arr)
```

```
## Find the data path
path(arr)

## Find the data path
dim(arr)
```

---

makeRandomGRanges *Creating random GRanges & GInteractions*

Description

Creating random GRanges & GInteractions

Usage

```
makeRandomGRanges(seqinfo, n = 100, ...)

makeRandomGInteractions(seqinfo, n = 100, interchromosomal = TRUE, ...)

## S4 method for signature 'Seqinfo'
makeRandomGRanges(seqinfo, n, .rows = NULL)

## S4 method for signature 'Seqinfo'
makeRandomGInteractions(seqinfo, n, interchromosomal)
```

Arguments

seqinfo	A Seqinfo object containing the chromosome names, lengths, and genome build.
n	Integer describing the number of random sequences to generate
...	Additional arguments.
interchromosomal	Boolean (TRUE/FALSE) indicating whether interchromosomal interactions should be allowed. Default is TRUE.
.rows	(internal use only) vector of row positions to sample from seqinfo.

Value

A GRanges or GInteractions object with ranges selected randomly with replacement on the provided seqinfo.

Examples

```
## Define Seqinfo containing chromosome info
if (require(TxDb.Hsapiens.UCSC.hg38.knownGene)) {
    txdb <- TxDb.Hsapiens.UCSC.hg38.knownGene
    si <- seqinfo(txdb)
    si <- GenomeInfoDb::keepStandardChromosomes(si)
} else {
    si <- Seqinfo(
        seqnames=c("chr1", "chr2"),
        seqlengths=rep(200e6, 2),
        genome="hg38"
    )
}

## Make some GRanges
set.seed(123)
makeRandomGRanges(si, 100)

## Make some GInteractions
set.seed(123)
makeRandomGInteractions(si, n=100)

## Make some GInteractions only on same chromosome
set.seed(123)
makeRandomGInteractions(si, n=100, interchromosomal=FALSE)

## Use specific binSizes
n <- 100
binOptions <- seq(5e3, 200e3, by=5e3)
si <- Seqinfo(seqnames="chr1", seqlengths=200e6, genome="hg38")
set.seed(123)
bins <- sample(binOptions, n, replace=TRUE)
makeRandomGInteractions(si, n) |>
    resize(bins) |>
    trim()
```

MatrixSelection-class *MatrixSelection Class*

Description

An object containing the selected indices of a matrix.

Value

A MatrixSelection object (see description)

Slots

`x` Vector of selected indices from a matrix of 'dim = buffer*2+1'.

`buffer` Integer indicating the buffer size, or number of pixels around a matrix.

Examples

```
selectCenterPixel(0, 5)
```

`MergedGInteractions-class`

MergedGInteractions Class

Description

The 'MergedGInteractions' class extends the 'GInteractions' to contain additional information about the pairs being merged.

Details

The 'MergedGInteractions' class uses a delegate object during initialization to assign its 'GInteractions' slots. In addition to containing information from all pairs, it also behaves as a 'GInteractions' object. 'mergePairs()' builds this object.

Value

A MergedGInteractions object (see description)

Slots

`delegate` A 'GInteractions' object used to initialize 'GInteractions'-specific slots. This is the mergedPairs set of interactions.

`ids` An integer vector of ids linking indices in the 'delegate' slot all pairs ('allPairs' slot). These indices are parallel to 'delegate'.

`allPairs` A 'data.table' containing all input pairs combined. Also contains all metadata for each pair and 1) the source of the file, 2) an id, 3) which chromosome pair it belongs to (i.e. 'grp'), and 4) the assigned cluster from 'dbscan' (i.e. 'clst').

`selectionMethod` Character describing which method was used to select the final pair from the cluster of merged pairs.

See Also

[InteractionSet::GInteractions]

Examples

```
## Load required packages
library(data.table, include.only="fread")

## Load marinerData
if (!require("marinerData", quietly = TRUE))
    BiocManager::install("marinerData")

## Reference BEDPE files (loops called with SIP)
bedpeFiles <- c(
    marinerData::FS_5kbLoops.txt(),
    marinerData::WT_5kbLoops.txt()
)
names(bedpeFiles) <- c("FS", "WT")

## Read in bedpeFiles as a list of GInteractions
## Use only first 1000 rows for fast example
giList <-
    lapply(bedpeFiles, fread, nrows=1000) |>
    lapply(as_ginteractions)

## Cluster & merge pairs
x <- mergePairs(x = giList,
                radius = 10e03,
                column = "APScoreAvg")

class(x)
```

`mergePairs` *Merge sets of paired interactions*

Description

Sets of paired range objects (i.e., 'GInteractions') are first clustered by genomic distance with 'dbscan', then a representative interaction is selected for each cluster.

Usage

```
mergePairs(
  x,
  radius,
  method = "manhattan",
  column = NULL,
  selectMax = TRUE,
  pos = "center"
)

## S4 method for signature 'list_OR_SimpleList_OR_GInteractions,numeric'
```

```
mergePairs(
  x,
  radius,
  method = "manhattan",
  column = NULL,
  selectMax = TRUE,
  pos = "center"
)
```

Arguments

x	List of 'GInteractions' or 'data.frame'-like objects.
radius	Numeric describing the distance in base pairs used to define a cluster or pairs.
method	Character describing the distance measure to be used. This must be one of "euclidean", "maximum", "manhattan", "canberra", "binary" or "minkowski". Any unambiguous substring can be given. Default is "manhattan".
column	Character denoting the column to be used to select among clustered interactions.
selectMax	Logical. TRUE (default) uses 'which.max()' to select the interaction pair. FALSE uses 'which.min()'. Only applicable when 'column' is specified.
pos	Positions used for clustering pairs. Must be one of "start", "end" or "center". Default is "center".

Details

Interactions are clustered into groups using the provided base pair 'radius', and distance 'method' with 'dbscan()'. Representative interactions are selected for each group by one of two methods. If 'column' and 'selectMax' arguments are provided, the representative interaction with the maximum (or minimum) value in 'column' is returned for each cluster. If these parameters are missing, new ranges for each pair are returned by calculating the median of modes for each cluster.

Value

Returns a 'MergedGInteractions' object.

Examples

```
## Load marinerData
if (!require("marinerData", quietly = TRUE))
    BiocManager::install("marinerData")

bedpeFiles <- c(
    marinerData::FS_5kbLoops.txt(),
    marinerData::WT_5kbLoops.txt()
)
names(bedpeFiles) <- c("FS", "WT")

## Read in bedpeFiles as a list of GInteractions
## Use only first 1000 rows for fast example
giList <-
```

```
    lapply(bedpeFiles, read.table, header=TRUE, nrows=1000) |>
    lapply(as_ginteractions)

## Cluster & merge pairs
x <- mergePairs(x = giList,
                radius = 10e03,
                column = "APScoreAvg")
x
```

---

path,InteractionMatrix-method
Accessor for h5File path from an InteractionMatrix

Description

Returns the file path describing where the on-disk HDF5 data associated with the InteractionMatrix object is stored.

This method circumvents the 'assays<-' and 'path<-' methods for updating the HDF5 path because they are not accessible when the file path is broken.

Usage

```
## S4 method for signature 'InteractionMatrix'
path(object)

## S4 replacement method for signature 'InteractionMatrix'
path(object) <- value
```

Arguments

object	InteractionMatrix object
value	String (length-one character vector) to use for path replacement.

Details

If the file no longer exists, the path is returned along with a warning.

This allows the file path to be updated even if the original linked data no longer exists.

Value

The path to the HDF5 file associated with the InteractionMatrix object.

Updates path to HDF5 file for the InteractionMatrix object.

Examples

```
## Load marinerData
if (!require("marinerData", quietly = TRUE))
    BiocManager::install("marinerData")

## Read .hic file paths
hicFiles <- c(
    marinerData::LEUK_HEK_PJA27_inter_30.hic(),
    marinerData::LEUK_HEK_PJA30_inter_30.hic()
)
names(hicFiles) <- c("FS", "WT")

#################################
## Accessing path to HDF5 data ##
#################################

## Create example interactions
x <- read.table(text="
        9 14000000 14500000 9 14500000 15000000
        9 89500000 90000000 9 89500000 90000000
        9 23500000 24000000 9 23500000 24000000")
x <- as_ginteractions(x)

## Extract 3 pixels from 2 hic files
imat <- pullHicPixels(x, hicFiles, 500e03)

## Access path
path(imat)

#################################
## Updating path to HDF5 data ##
#################################

## Create example interactions
x <- read.table(text="
        9 14000000 14500000 9 14500000 15000000
        9 89500000 90000000 9 89500000 90000000
        9 23500000 24000000 9 23500000 24000000")
x <- as_ginteractions(x)

## Extract 3 pixels from 2 hic files
h5File <- tempfile(fileext=".h5")
imat <- pullHicPixels(x, hicFiles, 500e03, h5File=h5File)

## Move file to new location
newFile <- tempfile(fileext="_new.h5")
file.rename(from=h5File, to=newFile)

## Update path
path(imat) <- newFile
path(imat)
```

pileupBoundaries *Pileup Hi-C contacts around boundary regions*

Description

pileupBoundaries expands input loci to the specified 'width', extracts then aggregates them into a single matrix. This can be used to aggregate windows of interactions centered on a set of loci.

Usage

```
pileupBoundaries(
  x,
  files,
  binSize,
  width = 5e+05,
  normalize = TRUE,
  FUN = sum,
  nBlocks = 50,
  verbose = TRUE,
  BPPARAM = bpparam(),
  blockSize = 1e+06,
  ...
)

## S4 method for signature 'GRanges_OR_GInteractions,character,numeric'
pileupBoundaries(
  x,
  files,
  binSize,
  width = 5e+05,
  normalize = TRUE,
  FUN = sum,
  nBlocks = 50,
  verbose = TRUE,
  BPPARAM = bpparam(),
  blockSize = 1e+06,
  ...
)
```

Arguments

x	GRanges or GInteractions object containing the loci to be aggregated. GInteractions will be split into unique anchors.
files	Character file paths to '.hic' files.
binSize	Integer (numeric) describing the resolution (range widths) of the paired data. Note that small values for this argument may lead to R session crashes.

width	Number of base pairs to expand the loci of interest in 'x'.
normalize	Boolean, whether to normalize the aggregated values to the number of interactions.
FUN	Function to use for aggregating.
nBlocks	Number of blocks for block-processing arrays. Default is 50. Increase this for large datasets. To read and process all data at once, set this value to 1.
verbose	Boolean (TRUE or FALSE) describing whether to report block-processing progress.
BPPARAM	Parallelization params (passed to 'BiocParallel::bplapply()'). Default is the result of 'BiocParallel::bpparams()'. Parallel processing is not available when 'by=interactions'.
blockSize	Number (length one numeric vector) describing the size in base-pairs to pull from each '.hic' file. Default is 1e6. For large '.hic' files 'blockSize' can be reduced to conserve the amount of data read in at a time. Larger 'blockSize' values speed up performance, but use more memory.
...	Additional arguments passed to 'pullHicMatrices()'.

Details

It may be necessary to adjust the 'zrange' in 'plotMatrix()' since the Hi-C diagonal will dominate the scale.

Using small 'binSize' values with large ranges may lead to pulling very large sections of a Hi-C map that can crash your R session. If this happens try increasing the 'binSize' and 'nBlocks' parameters, while lower the 'blockSize' parameter.

Value

A DelayedArray of aggregated counts.

Examples

```
## Load marinerData
if (!require("marinerData", quietly = TRUE))
    BiocManager::install("marinerData")

## Read .hic file paths
hicFile <- marinerData::LEUK_HEK_PJA27_inter_30.hic()
names(hicFile) <- "FS"

## Loops
loops <-
    marinerData::FS_5kbLoops.txt() |>
    read.table(header=TRUE, nrows=100) |>
    as_ginteractions() |>
    GenomeInfoDb::`seqlevelsStyle<-`(value='ENSEMBL')

## Warn about small binSize
pileupBoundaries(x=loops, files=hicFile, binSize=50e3)
```

pileupDomains *Pileup Hi-C domains*

Description

pileupDomains expands then extracts regions/domains from Hi-C files, regularizes them so they are the same size, then aggregates them into a single matrix. This can be used to perform aggregate TAD analysis.

Usage

```
pileupDomains(
  x,
  files,
  binSize,
  buffer = 0.5,
  ndim = c(100, 100),
  scale = TRUE,
  normalize = TRUE,
  FUN = sum,
  nBlocks = 50,
  verbose = TRUE,
  BPPARAM = bpparam(),
  blockSize = 1e+06,
  ...
)

## S4 method for signature 'GRanges_OR_GInteractions,character,numeric'
pileupDomains(
  x,
  files,
  binSize,
  buffer = 0.5,
  ndim = c(100, 100),
  scale = TRUE,
  normalize = TRUE,
  FUN = sum,
  nBlocks = 50,
  verbose = TRUE,
  BPPARAM = bpparam(),
  blockSize = 1e+06,
  ...
)
```

Arguments

x	GRanges or GInteractions object containing the TADs or Loops to be aggregated.

files	Character file paths to '.hic' files.
binSize	Integer (numeric) describing the resolution (range widths) of the paired data. Note that small values for this argument may lead to R session crashes.
buffer	Fraction (length one numeric vector) pair-distance to expand around the resulting range.
ndim	Numeric vector of length two describing the new dimensions of the output matrices.
scale	Boolean (TRUE/FALSE) indicating whether the values in the new matrices should be scaled to the total signal in each matrix.
normalize	Boolean, whether to normalize the aggregated values to the number of interactions.
FUN	Function to use for aggregating.
nBlocks	Number of blocks for block-processing arrays. Default is 50. Increase this for large datasets. To read and process all data at once, set this value to 1.
verbose	Boolean (TRUE or FALSE) describing whether to report block-processing progress.
BPPARAM	Parallelization params (passed to 'BiocParallel::bplapply()'). Default is the result of 'BiocParallel::bpparams()'. Parallel processing is not available when 'by=interactions'.
blockSize	Number (length one numeric vector) describing the size in base-pairs to pull from each '.hic' file. Default is 1e6. For large '.hic' files 'blockSize' can be reduced to conserve the amount of data read in at a time. Larger 'blockSize' values speed up performance, but use more memory.
...	Additional arguments passed to 'pullHicMatrices()'.

Details

It may be necessary to adjust the 'zrange' in 'plotMatrix()' since the Hi-C diagonal will dominate the scale.

If interactions are passed to the function, only intrachromosomal ranges are maintained.

Using small 'binSize' values with large ranges may lead to pulling very large sections of a Hi-C map that can crash your R session. If this happens try increasing the 'binSize' and 'nBlocks' parameters, while lower the 'blockSize' parameter.

Value

A DelayedArray of aggregated counts.

Examples

```
## Load marinerData
if (!require("marinerData", quietly = TRUE))
    BiocManager::install("marinerData")

## Read .hic file paths
hicFile <- marinerData::LEUK_HEK_PJA27_inter_30.hic()
names(hicFile) <- "FS"
```

```
## Loops
loops <-
    marinerData::FS_5kbLoops.txt() |>
    read.table(header=TRUE, nrows=100) |>
    as_ginteractions() |>
    GenomeInfoDb::`seqlevelsStyle<-`(value='ENSEMBL')

## Warn about small binSize
pileupDomains(x=loops, files=hicFile, binSize=50e3, buffer=0.25)
```

`pileupPixels` *Pileup Hi-C pixels*

Description

pileupPixels optionally removes short interactions that intersect the diagonal before extracting then aggregating square regions around each pixel from Hi-C files. This is also known as aggregate peak analysis (APA)

Usage

```
pileupPixels(
  x,
  files,
  binSize,
  buffer = 5,
  removeShort = TRUE,
  minPairDist = 0,
  normalize = TRUE,
  FUN = sum,
  nBlocks = 5,
  verbose = TRUE,
  BPPARAM = bpparam(),
  ...
)

## S4 method for signature 'GInteractions,character,numeric'
pileupPixels(
  x,
  files,
  binSize,
  buffer = 5,
  removeShort = TRUE,
  minPairDist = 0,
  normalize = TRUE,
  FUN = sum,
```

```
  nBlocks = 5,
  verbose = TRUE,
  BPPARAM = bpparam(),
  ...
)
```

Arguments

x	GInteractions object containing interactions to extract from Hi-C files. These should be pixels of a single ‘binSize‘ in width.
files	Character file paths to ‘.hic‘ files.
binSize	Integer (numeric) describing the resolution (range widths) of the paired data.
buffer	Integer indicating the buffer size, or number of pixels
removeShort	Boolean, whether to remove short pairs (Default) or not.
minPairDist	Pairs with a distance less than or equal to this value will be filtered out.
normalize	Boolean, whether to normalize the aggregated values to the number of interactions (after filtering out short pairs - if applicable).
FUN	Function to use for aggregating.
nBlocks	Number of blocks for block-processing arrays. Default is 5. Increase this for large datasets. To read and process all data at once, set this value to 1.
verbose	Boolean (TRUE or FALSE) describing whether to report block-processing progress.
BPPARAM	Parallelization params (passed to ‘BiocParallel::bplapply()‘). Default is the result of ‘BiocParallel::bpparams()‘. Parallel processing is not available when ‘by=interactions‘.
...	Additional arguments passed to ‘pullHicMatrices()‘.

Details

Note that pair distance filtering is done after expanding interactions to matrices.

Value

A DelayedMatrix of aggregated counts.

Examples

```
## Load marinerData
if (!require("marinerData", quietly = TRUE))
    BiocManager::install("marinerData")

## Read .hic file paths
hicFile <- marinerData::LEUK_HEK_PJA30_inter_30.hic()
names(hicFile) <- "WT"

## Loops
loops <-
    WT_5kbLoops.txt() |>
```

```
    setNames("WT") |>
    read.table(header=TRUE, nrows=1000) |>
    as_ginteractions(keep.extra.columns=FALSE) |>
    assignToBins(binSize=5e3)

## Removes the "chr" prefix for compatibility
## with the preprocessed hic files
GenomeInfoDb::seqlevelsStyle(loops) <- 'ENSEMBL'

## APA
mat <- pileupPixels(
    x=loops,
    files=hicFile,
    binSize=5e3,
    minPairDist=50e3,
    normalize=FALSE
)
```

`pixelsToMatrices` *Expand pixels to submatrices*

Description

Pixels are defined as paired-ranges with starts & ends equal to their 'binSize'. This function takes GInteractions fitting this description and expands the ranges such that there is a 'buffer' of pixels around each range.

Usage

```
pixelsToMatrices(x, buffer)

## S4 method for signature 'GInteractions,numeric'
pixelsToMatrices(x, buffer)
```

Arguments

`x`	GInteractions object.
`buffer`	Number (length one numeric vector) of pixels around the pixels in 'x'.

Details

For example, a buffer of 3 would return a GInteractions object with 3 pixels surrounding the original pixel ranges.

After using 'pullHicMatrices()', the result will return a matrix of row and column dimensions of buffer*2+1.

Note, this function does not handle out-of-bound ranges.

Value

'x' with updated ranges.

Examples

```
## Define example 100bp pixel
library(InteractionSet)
pixel <- GInteractions(
    anchor1=GRanges("chr1:500-600"),
    anchor2=GRanges("chr1:2000-2100")
)

## Expand pixel to matrix with
## 3 pixels surrounding the center
## pixel
region <- pixelsToMatrices(x=pixel, buffer=3)
region
```

`plotEnrichment` *Adjust loop enrichment to remove distance- dependent effect.*

Description

Adjust loop enrichment to remove distance- dependent effect.

Show diagnostic plot of loop enrichment before and after distance adjustment.

Usage

```
plotEnrichment(scores, interactions, k = 25, nknots = 10, plot = TRUE)

adjustEnrichment(x, interactions, k = 25, nknots = 10)

## S4 method for signature 'DelayedMatrix_OR_matrix,GInteractions'
adjustEnrichment(x, interactions, k = 25, nknots = 10)

## S4 method for signature 'numeric,GInteractions'
plotEnrichment(scores, interactions, k = 25, nknots = 10, plot = TRUE)
```

Arguments

`scores`	Numeric vector of enrichment scores.
`interactions`	A GInteractions Object containing the interactions used to calculate enrichment scores.
`k`	Number of observations for rolling window.
`nknots`	integer or function giving the number of knots to use see '?smooth.spline' for more info.

plot	Boolean (default=FALSE), of whether to show diagnostic plot.
x	A DelayedMatrix or matrix with enrichment scores.

Value

A DelayedMatrix of enrichment scores where rows are loops and columns are Hi-C files.

A plot (and associated data) for visualizing loop enrichment before and after distance adjustment.

Examples

```
## Load marinerData
if (!require("marinerData", quietly = TRUE))
    BiocManager::install("marinerData")

## Read .hic file paths
hicFiles <- c(
    marinerData::LEUK_HEK_PJA27_inter_30.hic(),
    marinerData::LEUK_HEK_PJA30_inter_30.hic()
)
names(hicFiles) <- c("FS", "WT")

## Read in loops as GInteractions object
loops <-
    WT_5kbLoops.txt() |>
    setNames("WT") |>
    read.table(header=TRUE, nrows=1000) |>
    as_ginteractions(keep.extra.columns=FALSE)

## Removes the "chr" prefix for compatibility
## with the preprocessed hic files
GenomeInfoDb::seqlevelsStyle(loops) <- 'ENSEMBL'

## Calculate loop enrichment
enrich <- calcLoopEnrichment(
    x=assignToBins(loops, 100e03),
    files=hicFiles
)

adjustEnrichment(enrich, loops)


plotEnrichment(enrich[,1], loops)
```

---

plotMatrix *Plot matrix*

Description

Used to plot single or aggregate matrix such as aggregate peak analysis.

Usage

```
plotMatrix(
  data,
  params = NULL,
  x = NULL,
  y = NULL,
  width = NULL,
  height = NULL,
  just = c("left", "top"),
  default.units = "inches",
  draw = TRUE,
  palette = colorRampPalette(RColorBrewer::brewer.pal(9, "YlGnBu")),
  zrange = NULL,
  na.color = "grey"
)

## S4 method for signature 'DelayedMatrix_OR_matrix'
plotMatrix(
  data,
  params = NULL,
  x = NULL,
  y = NULL,
  width = NULL,
  height = NULL,
  just = c("left", "top"),
  default.units = "inches",
  draw = TRUE,
  palette = colorRampPalette(RColorBrewer::brewer.pal(9, "YlGnBu")),
  zrange = NULL,
  na.color = "grey"
)
```

Arguments

data	'DelayedMatrix', 'matrix', list of matrices, or 3 column 'data.frame' of APA results.
params	Optional 'pgParams' object containing relevant function parameters.
x	Numeric or unit object specifying the x-location of plot.
y	Numeric or unit object specifying the y-location of plot.
width	Numeric or unit object specifying the width of plot.
height	Numeric or unit object specifying the height of plot.
just	String or numeric vector specifying the justification of the viewport relative to its (x, y) location.
default.units	String indicating the default units to use if 'x', 'y', 'width', or 'height' are only given as numeric vectors.
draw	Logical value indicating whether graphics output should be produced.

palette	'colorRampPalette' function to use for mapping values to colors.
zrange	Vector of length 2; max and min values to set color scale
na.color	String indicating the color to use for mapping NA values.

Value

Function will draw a color-mapped matrix and return an S3 object of class 'MatrixPlot'.

Examples

```
library(plotgardener)
library(RColorBrewer)

## Create divergent matrix ####
m <- matrix(data=rnorm(n=21*21, mean=0, sd=2), nrow=21, ncol=21)

## Define parameters
p <- pgParams(width=3, height=3, default.units="inches")

## Create page
pageCreate(params=p)

## Plot apa
plot <- plotMatrix(data=m,
                   x=p$width/2,
                   y=p$height/2,
                   width=p$width*0.5, height = p$width*0.5,
                   just=c("center", "center"),
                   palette=colorRampPalette(c("blue", "white", "red")),
                   zrange=NULL)

## Annotate legend
annoHeatmapLegend(plot=plot,
                  x=2.3,
                  y=0.75,
                  width=0.1,
                  height=0.75)


## Create sequential matrix
m <- matrix(data=sample(0:100, 21*21, replace=TRUE), nrow=21, ncol=21)

## Define parameters
p <- pgParams(width=3, height=3, default.units="inches")

## Create page
pageCreate(params=p)

## Plot apa
plot <- plotMatrix(data=m,
                   x=p$width/2,
                   y=p$height/2,
```

```
                    width=p$width*0.5,
                    height=p$width*0.5,
                    just=c("center", "center"),
                    palette=colorRampPalette(c("white", "dark red")),
                    zrange = NULL)

## Annotate legend
annoHeatmapLegend(plot=plot,
                  x=2.3,
                  y=0.75,
                  width=0.1,
                  height=0.75)
```

`pullHicMatrices` *Pull submatrices from '.hic' files*

Description

The dimensions of the pulled submatrix is defined by dividing the widths of anchors in 'x' by the 'binSize'. When the anchor widths are the same for each interaction, an InteractionArray is returned. However, if the anchor widths differ in 'x', an InteractionJaggedArray is returned instead.

Usage

```
pullHicMatrices(
  x,
  files,
  binSize,
  ...,
  h5File = tempfile(fileext = ".h5"),
  half = "both",
  norm = "NONE",
  matrix = "observed",
  blockSize = 248956422,
  onDisk = TRUE,
  compressionLevel = 0,
  chunkSize = 1
)

## S4 method for signature 'GInteractions,character,numeric'
pullHicMatrices(
  x,
  files,
  binSize,
  h5File,
  half,
  norm,
```

```
  matrix,
  blockSize,
  onDisk,
  compressionLevel,
  chunkSize
)
```

Arguments

x	GInteractions object containing interactions to extract from Hi-C files.
files	Character file paths to '.hic' files.
binSize	Integer (numeric) describing the resolution (range widths) of the paired data.
...	Additional arguments.
h5File	Character file path to save '.h5' file.
half	String (character vector of length one) indicating whether to keep values for the upper triangular ('half="upper"') where 'start1 < start2', lower triangular ('half="lower"') where 'start1 > start2', or both ('half="both"', default). When 'half="upper"' all lower triangular values are 'NA'. When 'half="lower"' all upper triangular values are 'NA'. When 'half="both"' there are no 'NA' values. For interchromosomal interactions there is no inherent directionality between chromosomes, so data is returned regardless of specified order.
norm	String (length one character vector) describing the Hi-C normalization to apply. Use 'strawr::readHicNormTypes()' to see accepted values for each file in 'files'.
matrix	String (length one character vector) Type of matrix to extract. Must be one of "observed", "oe", or "expected". "observed" is observed counts, "oe" is observed/expected counts, "expected" is expected counts.
blockSize	Number (length one numeric vector) describing the size in base-pairs to pull from each '.hic' file. Default is 248956422 (the length of the longest chromosome in the human hg38 genome). For large '.hic' files 'blockSize' can be reduced to conserve the amount of data read in at a time. Larger 'blockSize' values speed up performance, but use more memory.
onDisk	Boolean (length one logical vector that is not NA) indicating whether extracted data should be stored on disk in an HDF5 file. Default is TRUE.
compressionLevel	Number (length one numeric vector) between 0 (Default) and 9 indicating the compression level used on HDF5 file.
chunkSize	Number (length one numeric vector) indicating how many values of 'x' to chunk for each write to HDF5 stored data. This has downstream implications for accessing subsets later. For small 'compressionLevel' values use smaller 'chunkSize' values and for large 'compressionLevel' values use large (i.e. 'length(x)') values to improve performance.

Value

InteractionSet object with a 4-dimensional array of Hi-C submatrices, rownames, and colnames. Array is stored with the following dimensions: Interactions in 'x', Hi-C 'files', rows of submatrix,

columns of submatrix. The submatrices returned have rows cooresponding to anchor1 of ‘x‘ and columns correspond to anchor2 of ‘x‘.

Examples

```
## Load marinerData
if (!require("marinerData", quietly = TRUE))
    BiocManager::install("marinerData")

## Read .hic file paths
hicFiles <- c(
    marinerData::LEUK_HEK_PJA27_inter_30.hic(),
    marinerData::LEUK_HEK_PJA30_inter_30.hic()
)
names(hicFiles) <- c("FS", "WT")

## Read in loop pixels as GInteractions object
pixels <-
  WT_5kbLoops.txt() |>
  setNames("WT") |>
  read.table(header=TRUE) |>
  as_ginteractions(keep.extra.columns=FALSE) |>
  assignToBins(binSize=100e3)

## Removes the "chr" prefix for compatibility
## with the preprocessed hic files
GenomeInfoDb::seqlevelsStyle(pixels) <- 'ENSEMBL'

## Expand pixels to regions for pulling
## Hi-C submatrices
regions <- pixelsToMatrices(x=pixels, buffer=5)

## Extract 11x11 count matrices from the
## first 100 regions and 2 Hi-C files
iarr <- pullHicMatrices(x=regions[1:100],
                        files=hicFiles,
                        binSize=100e3)
iarr

## Access count matrices
counts(iarr)

## Display the start bin of each
## interaction in the count
## matrices
counts(iarr, showDimnames=TRUE)

## InteractionJaggedArray example
gi <- read.table(text="
            1 51000000 51300000 1 51000000 51500000
            2 52000000 52300000 3 52000000 52500000
            1 150000000 150500000 1 150000000 150300000
            2 52000000 52300000 2 52000000 52800000") |>
```

```
    as_ginteractions()

iarr <- pullHicMatrices(gi, hicFiles, 100e03, half="both")
iarr

counts(iarr)
```

---

pullHicPixels *Pull contact frequency from '.hic' files*

Description

Pull contact frequency from '.hic' files

Usage

```
pullHicPixels(
  x,
  files,
  binSize,
  ...,
  h5File = tempfile(fileext = ".h5"),
  half = "both",
  norm = "NONE",
  matrix = "observed",
  blockSize = 248956422,
  onDisk = TRUE,
  compressionLevel = 0,
  chunkSize = 1
)

## S4 method for signature 'GInteractions,character,numeric'
pullHicPixels(
  x,
  files,
  binSize,
  h5File,
  half,
  norm,
  matrix,
  blockSize,
  onDisk,
  compressionLevel,
  chunkSize
)
```

Arguments

x	GInteractions object containing interactions to extract from Hi-C files.
files	Character file paths to '.hic' files.
binSize	Integer (numeric) describing the resolution (range widths) of the paired data.
...	Additional arguments.
h5File	Character file path to save '.h5' file.
half	String (character vector of length one) indicating whether to keep values for the upper triangular ('half="upper"') where 'start1 < start2', lower triangular ('half="lower"') where 'start1 > start2', or both ('half="both"', default). When 'half="upper"' all lower triangular values are 'NA'. When 'half="lower"' all upper triangular values are 'NA'. When 'half="both"' there are no 'NA' values. For interchromosomal interactions there is no inherent directionality between chromosomes, so data is returned regardless of specified order.
norm	String (length one character vector) describing the Hi-C normalization to apply. Use 'strawr::readHicNormTypes()' to see accepted values for each file in 'files'.
matrix	String (length one character vector) Type of matrix to extract. Must be one of "observed", "oe", or "expected". "observed" is observed counts, "oe" is observed/expected counts, "expected" is expected counts.
blockSize	Number (length one numeric vector) describing the size in base-pairs to pull from each '.hic' file. Default is 248956422 (the length of the longest chromosome in the human hg38 genome). For large '.hic' files 'blockSize' can be reduced to conserve the amount of data read in at a time. Larger 'blockSize' values speed up performance, but use more memory.
onDisk	Boolean (length one logical vector that is not NA) indicating whether extracted data should be stored on disk in an HDF5 file. Default is TRUE.
compressionLevel	Number (length one numeric vector) between 0 (Default) and 9 indicating the compression level used on HDF5 file.
chunkSize	Number (length one numeric vector) indicating how many values of 'x' to chunk for each write to HDF5 stored data. This has downstream implications for accessing subsets later. For small 'compressionLevel' values use smaller 'chunkSize' values and for large 'compressionLevel' values use large (i.e. 'length(x)') values to improve performance.

Value

InteractionSet object with a 2-dimensional array of Hi-C interactions (rows) and Hi-C sample (columns).

Examples

```
## Load marinerData
if (!require("marinerData", quietly = TRUE))
    BiocManager::install("marinerData")

## Read .hic file paths
```

```
hicFiles <- c(
    marinerData::LEUK_HEK_PJA27_inter_30.hic(),
    marinerData::LEUK_HEK_PJA30_inter_30.hic()
)
names(hicFiles) <- c("FS", "WT")

## Read in loop pixels as GInteractions object
pixels <-
  WT_5kbLoops.txt() |>
  setNames("WT") |>
  read.table(header=TRUE) |>
  as_ginteractions(keep.extra.columns=FALSE) |>
  assignToBins(binSize=100e3)

## Removes the "chr" prefix for compatibility
## with the preprocessed hic files
GenomeInfoDb::seqlevelsStyle(pixels) <- 'ENSEMBL'

## Extract the first 100 pixels
imat <- pullHicPixels(x=pixels[1:100],
                      files=hicFiles,
                      binSize=100e3)
imat

## Access count matrix
counts(imat)
```

`regularize` *Regularize JaggedArray or InteractionJaggedArray objects*

Description

InteractionJaggedArray objects and their count matrices (JaggedArray objects) contain variable dimension matrices. The ‘regularize‘ function resizes these matrices to the new dimensions supplied in ‘ndim‘. The result is a DelayedArray object (for JaggedArray) or an InteractionArray object (for InteractionJaggedArray).

Usage

```
regularize(
  x,
  ndim = c(10, 10),
  h5File = tempfile(fileext = ".h5"),
  scale = TRUE,
  nBlocks = 5,
  verbose = TRUE,
  chunkSize = 1,
  compressionLevel = 0,
```

```
  ...
)

## S4 method for signature 'JaggedArray'
regularize(
  x,
  ndim,
  h5File,
  scale,
  nBlocks,
  verbose,
  chunkSize,
  compressionLevel
)

## S4 method for signature 'InteractionJaggedArray'
regularize(
  x,
  ndim,
  h5File,
  scale,
  nBlocks,
  verbose,
  chunkSize,
  compressionLevel
)
```

Arguments

x	A JaggedArray or InteractionJaggedArray object.
ndim	Numeric vector of length two describing the new dimensions of the output matrices.
h5File	Character file path to save '.h5' file.
scale	Boolean (TRUE/FALSE) indicating whether the values in the new matrices should be scaled to the total signal in each matrix.
nBlocks	Number of blocks for block-processing JaggedArrays. Default is 5. Increase this for large datasets. To read and process all data at once, set this value to 1.
verbose	Boolean (TRUE or FALSE) describing whether to report block-processing progress.
chunkSize	Number (length one numeric vector) indicating how many values of 'x' to chunk for each write to HDF5 stored data. This has downstream implications for accessing subsets later. For small 'compressionLevel' values use smaller 'chunkSize' values and for large 'compressionLevel' values use large (i.e. 'length(x)') values to improve performance.
compressionLevel	Number (length one numeric vector) between 0 (Default) and 9 indicating the compression level used on HDF5 file.
...	Additional arguments.

Details

Note that the interaction/binSize/count matrices relationship will be inconsistent in the resulting InteractionArray object and the row/col names will not be available.

Value

If ‘x‘ is a JaggedArray then ‘regularize‘ returns an HDF5-backed 4-dimensional DelayedArray object where the first and second dimensions are the rows and columns of the count matrices (‘ndim‘), the third dimension is the number of interactions and the fourth dimension is the number of files. If ‘x‘ is an InteractionJaggedArray then an InteractionArray object is returned where counts returns the object described above.

Examples

```
## Load marinerData
if (!require("marinerData", quietly = TRUE))
    BiocManager::install("marinerData")

## Read .hic file paths
hicFiles <- c(
    LEUK_HEK_PJA27_inter_30.hic(),
    LEUK_HEK_PJA30_inter_30.hic()
)
names(hicFiles) <- c("FS", "WT")

## Create test interactions
gi <- read.table(text="
            1 51000000 51300000 1 51000000 51500000
            2 52000000 52300000 3 52000000 52500000
            1 150000000 150500000 1 150000000 150300000
            2 52000000 52300000 2 52000000 52800000") |>
    as_ginteractions()
gi <- c(gi,gi) # make more interactions

## InteractionJaggedArray object
ija <- pullHicMatrices(gi, hicFiles, 100e03, half="both")

## Regularize InteractionJaggedArray
ia <- regularize(ija, ndim=c(5,5), nBlocks=1)
aggHicMatrices(ia, nBlocks=1)

## Regularize JaggedArray
ja <- counts(ija)
regularize(ja, ndim=c(5,5), nBlocks=1)
```

`removeShortPairs` *Remove interactions that would cross the Hi-C diagonal or a specified distance from the diagonal.*

Description

Removes short interactions with some padding from the diagonal. If you are resizing the regions with a function like 'pixelsToMatrices()', make sure this function is run afterwards.

Usage

```
removeShortPairs(x, padding = 0)

## S4 method for signature 'GInteractions'
removeShortPairs(x, padding = 0)
```

Arguments

x	A GInteractions object.
padding	Minimum distance away from the diagonal.

Details

Note this is only applies to intrachromosomal pairs, as pair distance is meaningless for interchromosomal pairs. Therefore, all interchromosomal pairs are kept.

Value

A GInteractions object with the short pairs removed.

Examples

```
## Example GInteractions object
gi <- as_ginteractions(read.table(
    text="
        seqnames1 start1 end1 seqnames2 start2 end2 keep
        chr1 300 400 chr1 300 400 'no'
        chr1 100 200 chr1 300 400 'yes'
        chr1 300 400 chr1 100 200 'yes'
        chr1 300 400 chr2 300 400 'yes'
        chr1 250 350 chr1 300 400 'only_with_padding_50'
        chr1 300 400 chr1 250 350 'only_with_padding_50'
        ",
    header=TRUE
))

## Remove pairs that would cross the diagonal
removeShortPairs(gi)

## Add 50bp of padding
removeShortPairs(gi, padding=50)
```

selectionMethod *Get selectionMethod from MergedGInteractions object*

Description

Get selectionMethod from MergedGInteractions object

Usage

```
selectionMethod(x, ...)

## S4 method for signature 'MergedGInteractions'
selectionMethod(x, ...)
```

Arguments

x	MergedGInteractions object.
...	Additional arguments.

Value

A character vector describing which selection method was used for merging.

Examples

```
## Load required packages
library(data.table, include.only="fread")

## Load marinerData
if (!require("marinerData", quietly = TRUE))
    BiocManager::install("marinerData")

## Reference BEDPE files (loops called with SIP)
bedpeFiles <- c(
    marinerData::FS_5kbLoops.txt(),
    marinerData::WT_5kbLoops.txt()
)
names(bedpeFiles) <- c("FS", "WT")

## Read in bedpeFiles as a list of GInteractions
## Use only first 1000 rows for fast example
giList <-
    lapply(bedpeFiles, fread, nrows=1000) |>
    lapply(as_ginteractions)

## Cluster & merge pairs
x <- mergePairs(x = giList,
                radius = 10e03,
                column = "APScoreAvg")
```

```
selectionMethod(x)
```

`selectPixel` *Get the pixel representing the strongest or weakest interaction in an InteractionArray*

Description

Get the pixel representing the strongest or weakest interaction in an InteractionArray

Usage

```
selectPixel(
  x,
  aggFUN = sum,
  selectFUN = "which.max",
  nBlocks = 5,
  verbose = TRUE
)

## S4 method for signature 'InteractionArray'
selectPixel(
  x,
  aggFUN = sum,
  selectFUN = "which.max",
  nBlocks = 5,
  verbose = TRUE
)
```

Arguments

`x`	InteractionArray object
`aggFUN`	Function to use for aggregating across Hi-C files. Must be passable to ‘which.max‘ or ‘which.min‘. Default is "sum".
`selectFUN`	Function to use for selecting among aggregated interactions. Must be one of "which.max" or "which.min".
`nBlocks`	Number of blocks for block-processing arrays. Default is 5. Increase this for large datasets. To read and process all data at once, set this value to 1.
`verbose`	Boolean (TRUE or FALSE) describing whether to report block-processing progress. Default is TRUE.

Value

A GInteractions object with the updated pixel interactions, along with a column with the aggregated max/min value for that pixel.

Examples

```
## Load marinerData
if (!require("marinerData", quietly = TRUE))
    BiocManager::install("marinerData")

## Read .hic file paths
hicFiles <- c(
    marinerData::LEUK_HEK_PJA27_inter_30.hic(),
    marinerData::LEUK_HEK_PJA30_inter_30.hic()
)
names(hicFiles) <- c("FS", "WT")

## Read in loops as GInteractions object
loops <-
    WT_5kbLoops.txt() |>
    setNames("WT") |>
    read.table(header=TRUE) |>
    as_ginteractions(keep.extra.columns=FALSE)

## Removes the "chr" prefix for compatibility
## with the preprocessed hic files
GenomeInfoDb::seqlevelsStyle(loops) <- 'ENSEMBL'

## Rebin loops to 2.5e6 resolution
loops <- assignToBins(x=loops, binSize=2.5e06)

## Pull 5x5 matrices
iarr <- pullHicMatrices(x=loops[1:5],
                        files=hicFiles,
                        binSize=500e3,
                        norm="KR",
                        half='upper')

## Select pixel
selectPixel(iarr)
```

`selectRadius` *Visualize selection for a MatrixSelection object*

Description

Note: that buffer must be the same as the selection functions to work appropriately

For ‘selectCoordinates‘, ‘rowInd‘ and ‘colInd‘ are paired such that the selected position in the matrix is ‘c(rowInd[1:i], colInd[1:j])‘ for ‘i‘ rows and ‘j‘ columns.

Usage

```
selectRadius(x, buffer, invert = FALSE)
```

```
selectCenterPixel(mhDist, buffer, invert = FALSE)

selectSubmatrix(m, invert = FALSE)

selectCoordinates(rowInd, colInd, buffer, invert = FALSE)

selectBlock(rowInd, colInd, buffer, invert = FALSE)

selectTopLeft(n, buffer, inset = 0, invert = FALSE)

selectTopRight(n, buffer, inset = 0, invert = FALSE)

selectBottomRight(n, buffer, inset = 0, invert = FALSE)

selectBottomLeft(n, buffer, inset = 0, invert = FALSE)

selectCorners(n, buffer, inset = 0, invert = FALSE)

selectRows(rows, buffer, invert = FALSE)

selectCols(cols, buffer, invert = FALSE)

selectInner(n, buffer, invert = FALSE)

selectOuter(n, buffer, invert = FALSE)

## S4 method for signature 'MatrixSelection'
show(object)

## S4 method for signature 'numeric'
selectRadius(x, buffer, invert = FALSE)

## S4 method for signature 'numeric'
selectCenterPixel(mhDist, buffer, invert = FALSE)

## S4 method for signature 'matrix'
selectSubmatrix(m, invert = FALSE)

## S4 method for signature 'numeric'
selectCoordinates(rowInd, colInd, buffer, invert = FALSE)

## S4 method for signature 'numeric'
selectBlock(rowInd, colInd, buffer, invert = FALSE)

## S4 method for signature 'numeric'
selectTopLeft(n, buffer, inset = 0, invert = FALSE)
```

```
## S4 method for signature 'numeric'
selectTopRight(n, buffer, inset = 0, invert = FALSE)

## S4 method for signature 'numeric'
selectBottomRight(n, buffer, inset = 0, invert = FALSE)

## S4 method for signature 'numeric'
selectBottomLeft(n, buffer, inset = 0, invert = FALSE)

## S4 method for signature 'numeric'
selectCorners(n, buffer, inset = 0, invert = FALSE)

## S4 method for signature 'numeric'
selectRows(rows, buffer, invert = FALSE)

## S4 method for signature 'numeric'
selectCols(cols, buffer, invert = FALSE)

## S4 method for signature 'numeric'
selectInner(n, buffer, invert = FALSE)

## S4 method for signature 'numeric'
selectOuter(n, buffer, invert = FALSE)
```

Arguments

x	Integer vector of manhattan distances to select.
buffer	Integer describing the number of pixels surrounding the central pixel.
invert	Boolean indicating whether to invert the selection.
mhDist	Integer vector of manhattan distances to select along with center pixel.
m	matrix with 1's indicating selected positions and 0's indicated unselected positions.
rowInd	Integer describing the row indices.
colInd	Integer describing the column indices.
n	Integer describing the number of outer pixels to select. Must be length of one.
inset	Integer describing the number of pixels to inset the selection from the outer edge of the matrix. Default of 0 uses no inset.
rows	Integer describing which rows to select.
cols	Integer describing which cols to select.
object	A MatrixSelection object.

Value

A text-based visualization of the select matrix indices.

Numeric vector of matrix indices (byRow).

Numeric vector of matrix indices (byRow).

Numeric vector of matrix indices (byRow).

Numeric vector of matrix indices (byRow).

Numeric vector of matrix indices (byRow).

Numeric vector of matrix indices (byRow).

Numeric vector of matrix indices (byRow).

Numeric vector of matrix indices (byRow).

Numeric vector of matrix indices (byRow).

Numeric vector of matrix indices (byRow).

Numeric vector of matrix indices (byRow).

Numeric vector of matrix indices (byRow).

Numeric vector of matrix indices (byRow).

Numeric vector of matrix indices (byRow).

Examples

```
res <- selectCenterPixel(0, 3)
show(res)
selectRadius(x=c(2,3,4), buffer=5, invert=FALSE)
selectCenterPixel(0, 5)
selectSubmatrix(m = matrix(rep(c(1,0,1), 3), nrow=3, ncol=3))
selectCoordinates(rowInd=1:3, colInd=1:3, buffer=5)
selectBlock(rowInd=1:3, colInd=1:3, buffer=5)
selectTopLeft(n=3, buffer=5, inset=1, invert=FALSE)
selectTopRight(n=3, buffer=5, inset=1, invert=FALSE)
selectBottomRight(n=3, buffer=5, inset=1, invert=FALSE)
selectBottomLeft(n=3, buffer=5, inset=1, invert=FALSE)
selectCorners(n=3, buffer=5, inset=1, invert=FALSE)
selectRows(rows=1:3, buffer=5, invert=FALSE)
selectCols(cols=1:3, buffer=5, invert=FALSE)
selectInner(n=1, buffer=5, invert=FALSE)
selectOuter(n=1, buffer=5, invert=FALSE)
```

---

seqnames1 *Access each portion of a GInteractions-like object*

Description

Access each portion of a GInteractions-like object

Usage

```
seqnames1(x, ...)

seqnames2(x, ...)

start1(x, ...)

end1(x, ...)

start2(x, ...)

end2(x, ...)

## S4 method for signature 'GInteractions_OR_InteractionSet'
seqnames1(x)

## S4 method for signature 'GInteractions_OR_InteractionSet'
seqnames2(x)

## S4 method for signature 'GInteractions_OR_InteractionSet'
start1(x)

## S4 method for signature 'GInteractions_OR_InteractionSet'
end1(x)

## S4 method for signature 'GInteractions_OR_InteractionSet'
start2(x)

## S4 method for signature 'GInteractions_OR_InteractionSet'
end2(x)
```

Arguments

x	GInteractions object.
...	Additional arguments.

Value

A vector of values corresponding to the requested component of a GInteractions-like object. For seqnames1 and seqnames2 the RLE is coerced to a character vector.

Examples

```
library(InteractionSet)
## Create example reference interactions objects
gi <- read.table(text="
    chr1 10 20 chr1 50 60
    chr2 30 40 chr2 60 70
    chr1 50 60 chr3 10 20") |>
```

```
    as_ginteractions()

iset <- InteractionSet(assays=matrix(nrow=3),
                       interactions=gi)

## Access vectors of values
seqnames1(gi)
start1(gi)
end1(gi)
seqnames2(gi)
start2(gi)
end2(gi)

## Also works for InteractionSet-like objects
seqnames1(iset)
start1(iset)
end1(iset)
seqnames2(iset)
start2(iset)
end2(iset)
```

---

sets *Get each set from a MergedGInteractions object*

Description

Returns the subset of MergedGInteractions that belong to each input source object (see these with ‘sources(x)‘). If the source pairs all come from the same object, their corresponding merged pair is returned. However, if at least one source pair comes from a different object, then that merged pair is not returned.

Usage

```
sets(x, include, exclude)

## S4 method for signature 'MergedGInteractions,missing,missing'
sets(x)

## S4 method for signature 'MergedGInteractions,character_OR_missing,missing'
sets(x, include)

## S4 method for signature 'MergedGInteractions,missing,character_OR_missing'
sets(x, exclude)

## S4 method for signature
## 'MergedGInteractions,character_OR_missing,character_OR_missing'
sets(x, include, exclude)
```

Arguments

x	MergedGInteractions object.
include	(Optional) A character vector of sources in which a pair must be present. For a list of available sources use ‘sources(x)‘.
exclude	(Optional) A character vector of sources in which a pair must be absent. For a list of available sources use ‘sources(x)‘.

Details

Optional ‘include‘ and ‘exclude‘ parameters modulate the behavior of ‘sets‘ to return different subsets of originating pairs. For example, ‘include‘ requires that the returned pairs be present in specific sources, while ‘exclude‘ requires that returned pairs be absent from specific sources. Sources not listed in either ‘include‘ or ‘exclude‘ are ignored (they may or may not) be present in the returned ‘MergedGInteractions‘ object. ‘include‘ and ‘exclude‘ can be used indepedently or in combination to return every possible set. If any of the same sources are used in both ‘include‘ and ‘exclude‘ the function will return a 0-length MergedGInteractions object.

Value

A list of subsetted ‘MergedGInteractions‘ objects or a ‘MergedGInteractions‘ object (if ‘include‘ and/or ‘exclude‘ are used).

Examples

```
## Load required packages
library(GenomicRanges)
library(InteractionSet)

## Define example anchor regions
gr1 <-
    GRanges(seqnames = "chr1",
            ranges = IRanges(start = c(30,40,40,70,80),
                             end = c(40,50,50,80,90)))
gr2 <-
    GRanges(seqnames = "chr1",
            ranges = IRanges(start = c(30,30,50,10,30),
                             end = c(40,40,60,20,40)))

## Form GInteractions and split into two files
giList <- split(x = GInteractions(gr1, gr2),
                f = c(rep(1,3), rep(2,2)))

## Merge pairs
x <- mergePairs(x = giList, radius = 20)

sets(x)
```

shiftRanges *Flexibly shifting GRanges according to strand*

Description

Flexibly shifting GRanges according to strand

Usage

```
shiftRanges(x, pos)

## S4 method for signature 'GRanges,character_OR_numeric'
shiftRanges(x, pos)
```

Arguments

x	GRanges object
pos	Position within anchors to resize the bin. Can be a character or integer vector of length 1 or ‘length(x)‘ designating the position for each element in bedpe. Character options are "start", "end" and "center". Integers are referenced from the start position for ’+’ and ’*’ strands and from the end position for the ’-’ strand.

Value

GRanges object with a single position range that has been shifted appropriately.

Examples

```
library(GenomicRanges)

## Create example GRanges
gr1 <- GRanges(seqnames = "chr1",
               ranges = IRanges::IRanges(start = rep(5000,3),
                                         end = rep(6000,3)),
               strand = c('+', '-', '*'))

gr2 <- gr1 |> promoters(upstream = 2000, downstream = 200)

## Shifting anchors by keyword
shiftRanges(gr1, 'start')
shiftRanges(gr1, 'end')
shiftRanges(gr1, 'center')

## Shifting anchors by position
shiftRanges(gr1, 100)
shiftRanges(gr1, c(100, 200, 300))

## Shifting back to TSS
```

```
shiftRanges(gr2, 2000)
```

---

snapToBins *Snap GRanges or GInteractions to nearest bins*

Description

Snap GRanges or GInteractions to nearest bins

Snap paired-objects to nearest bins

Usage

```
snapToBins(x, binSize)

## S4 method for signature 'GRanges,numeric'
snapToBins(x, binSize)

## S4 method for signature 'GInteractions,numeric'
snapToBins(x, binSize)
```

Arguments

x	'GInteractions' object.
binSize	Integer (numeric) describing the new size of each range.

Value

GRanges object snapped to the nearest 'binSize'.

Input object snapped to the nearest 'binSize'.

Examples

```
library(GenomicRanges)
## Example GRanges object
x <- GRanges(seqnames = c("chr1"),
             ranges = IRanges(start = c(1, 1, 25, 19, 21),
                              end = c(15, 11, 31, 31, 39)))

snapToBins(x, binSize = 5)
snapToBins(x, binSize = 10)
snapToBins(x, binSize = 20)

library(InteractionSet)
## Sample GInteractions object
x <- GInteractions(anchor1 = c(GRanges("chr1:1-15"),
                               GRanges("chr1:1-11")),
                   anchor2 = c(GRanges("chr1:25-31"),
```

```
                                GRanges("chr1:19-31")))

snapToBins(x, binSize = 5)
snapToBins(x, binSize = 10)
snapToBins(x, binSize = 20)
```

`sources` *Accessor for sources*

Description

Access the names or source files of a 'MergedGInteractions' object.

Usage

```
sources(x)

## S4 method for signature 'MergedGInteractions'
sources(x)
```

Arguments

x	MergedGInteractions object.

Value

A character vector of names or source files of a 'MergedGInteractions' object.

Examples

```
## Load required packages
library(data.table, include.only="fread")

## Load marinerData
if (!require("marinerData", quietly = TRUE))
    BiocManager::install("marinerData")

## Reference BEDPE files (loops called with SIP)
loopFiles <- c(
    marinerData::FS_5kbLoops.txt(),
    marinerData::WT_5kbLoops.txt()
)
names(loopFiles) <- c("FS", "WT")

## Read in loopFiles as a list of GInteractions
## Use only first 1000 rows for fast example
giList <-
    lapply(loopFiles, fread, nrows=1000) |>
    lapply(as_ginteractions)
```

```
## Cluster & merge pairs
x <- mergePairs(x = giList,
                radius = 10e03)

sources(x)
```

Index