Package 'LoopRig'

May 7, 2026

Title Integration and Analysis of Chromatin Loop Data

Version 0.1.1

Date 2019-12-04

Author Hassaan Maan [aut, cre]

Maintainer Hassaan Maan <hurmaan99@gmail.com>

Description

Common coordinate-based workflows involving processed chromatin loop and genomic element data are considered and packaged into appropriate customizable functions. Includes methods for linking element sets via chromatin loops and creating consensus loop datasets.

Depends R (>= 3.4.0)

Imports GenomicRanges, IRanges, utils, S4Vectors

Suggests testthat, knitr, rmarkdown, covr

VignetteBuilder knitr

LazyData true

License GPL-3 | file LICENSE

RoxygenNote 6.1.1

NeedsCompilation no

Repository CRAN

Date/Publication 2019-12-20 13:40:09 UTC

Contents

ConsensusLoops *Subset an object of class LoopRanges using consensus options*

Description

Performs filtering on looping data in LoopRanges objects based on custom parameters and returns a single GRangesList object indicating one looping dataset with two anchors

Usage

```
ConsensusLoops(loop_ranges, stringency = 1, overlap_threshold = 1,
  split_anchors = FALSE, resolutions = NULL, keep_all = FALSE)
```

Arguments

loop_ranges	An object of 'LoopRanges' class created from the LoopsToRanges() function
stringency	Integer (n>=0) indicating the number of looping datasets a loop from a given dataset must overlap with to be considered a consensus loop
overlap_threshold	Single numerical input in either percentage (0<=n<=1) overlap format if split_anchors = TRUE, or in base pair number format (n>=0) in split_anchors=FALSE (default=1)
split_anchors	A boolean (TRUE/FALSE) that determines if the different loop anchor sizes are considered together (default=TRUE) or separately (FALSE)
resolutions	An optional numerical vector of anchor sizes - to be used only when split_anchors=TRUE
keep_all	If TRUE, keeps all of the loops (concatenation of looping datasets)

Value

A 'LoopRanges' class object for the consensus loops

Examples

```
# Load loops into LoopRanges object
ovary_loops <- system.file("extdata/loops", "ovary_hg19.bedpe",
package = "LoopRig", mustWork = TRUE)
spleen_loops <- system.file("extdata/loops", "spleen_hg19.bedpe",
package = "LoopRig", mustWork = TRUE)
pancreas_loops <- system.file("extdata/loops", "pancreas_hg19.bedpe",
package = "LoopRig", mustWork = TRUE)
loops <- LoopsToRanges(ovary_loops, spleen_loops, pancreas_loops, custom_cols = 0)

# Determine consensus loops based on overlap of 1 bp in at least 2 datasets
ConsensusLoops(loops, stringency = 2, overlap_threshold = 1)
```

DropLoops *Drop loops from* LoopRanges *objects using anchor and loop sizes*

Description

Subset loops based on loop and anchor size filters. Can be used to filter the loops in the *LoopRanges* object before or after calling ConsensusLoops

Usage

```
DropLoops(loop_ranges, type = NULL, size = NULL)
```

Arguments

loop_ranges	A *LoopRanges* class object
type	A string indicating the type of filtering when determining which loops to drop: • "anchor_size" - Drops loops based on given anchor sizes • "loop_size" - Drops loops based on end-to-end loop size
size	A numerical vector indicating size range to keep (e.g. c(start, end))

Value

A subsetted *LoopRanges* class object

Examples

```
# Load loops into LoopRanges object
ovary_loops <- system.file("extdata/loops", "ovary_hg19.bedpe",
package = "LoopRig", mustWork = TRUE)
loops_ovary <- LoopsToRanges(ovary_loops, custom_cols = 0)

# Subset loops based on total length between 100 to 100000 bp
DropLoops(loops_ovary, type = "loop_size", size = c(100, 100000))

# Subset loops based on anchor size between 1000 to 25000 bp
DropLoops(loops_ovary, type = "anchor_size", size = c(1000, 25000))
```

`ElementsToRanges` *Create a list of ranges objects from element data*

Description

Uses tab delimited element data in the form of BED4 or BED12 files to create GRanges element objects

Usage

```
ElementsToRanges(..., element_names = NULL, custom_cols = NULL,
  custom_mcols = NULL)
```

Arguments

`...`	Any number of tab delimited element data files in BED4 or BED12 format
`element_names`	A character vector of names for the element datasets (optional)
`custom_cols`	An integer indicating the number of extra columns in the BED file.
`custom_mcols`	An integer or vector of integers indicating which columns are used for metadata (optional)

Value

An *ElementRanges* class object: list of GRanges element data objects

Examples

```
# Load enhancer and promoter elements into an ElementRanges object
enhancers <- system.file("extdata/elements", "enhancers.bed", package = "LoopRig", mustWork = TRUE)
promoters <- system.file("extdata/elements", "promoters.bed", package = "LoopRig", mustWork = TRUE)
element_ranges <- ElementsToRanges(enhancers, promoters,
element_names = c("enhancers", "promoters"),
custom_cols = 1, custom_mcols = 4)
element_ranges
```

`ExportBED` *Export a LoopRanges or* italics*ElementRanges object to a BED/BEDPE file*

Description

Uses ElementRanges objects to output to BED format, and LoopRanges objects to output to BEDPE format

Usage

```
ExportBED(obj, index = NULL, mcol = FALSE, file_name)
```

Arguments

obj	An object of *italics*LoopRanges or *italics*ElementRanges class
index	List index of LoopRanges or ElementRanges object to output
mcol	A boolean specifying whether the first mcol of the object are to be output (default=FALSE)
file_name	A string indicating the name and save location of the BED/BEDPE file

Examples

```
# Load enhancer and promoter elements into an ElementRanges object
enhancers <- system.file("extdata/elements", "enhancers.bed", package = "LoopRig", mustWork = TRUE)
promoters <- system.file("extdata/elements", "promoters.bed", package = "LoopRig", mustWork = TRUE)
element_ranges <- ElementsToRanges(enhancers, promoters,
element_names = c("enhancers", "promoters"),
custom_cols = 1, custom_mcols = 4)

# Call temporary directory
tempdir <- tempdir()

# Export enhancers into temporary directory
ExportBED(element_ranges, index = 1, file_name = paste(tempdir, "promoters_ex.bed", sep = "/"))

# Load loops into LoopRanges object
ovary_loops <- system.file("extdata/loops", "ovary_hg19.bedpe",
package = "LoopRig", mustWork = TRUE)
spleen_loops <- system.file("extdata/loops", "spleen_hg19.bedpe",
package = "LoopRig", mustWork = TRUE)
pancreas_loops <- system.file("extdata/loops", "pancreas_hg19.bedpe",
package = "LoopRig", mustWork = TRUE)
loops <- LoopsToRanges(ovary_loops, spleen_loops, pancreas_loops, custom_cols = 0)

# Export ovary loops into temporary directory
ExportBED(loops, index = 1, file_name = paste(tempdir, "ovary_loops_ex.bed", sep = "/"))
```

LinkedElements *Determines elements linked by loop anchors*

Description

Using a finalized *LoopRanges* loops object and two sets of elements, determines which elements are linked by loops through overlap of anchor regions

Usage

```
LinkedElements(loop_ranges, element_ranges_x, element_ranges_y,
  range_out_x = FALSE, range_out_y = FALSE, overlap_threshold = 1)
```

Arguments

loop_ranges	A single *LoopRanges* object that has been subset to one range through ConsensusLoops().
element_ranges_x	A subset *ElementRanges* class object. Subset appropriate ranges by using list indexing.
element_ranges_y	A subset *ElementRanges* class object. Subset appropriate ranges by using list indexing.
range_out_x	A boolean indicating if the output should be a subset element ranges object for element_ranges_x instead of the default dataframe. (default = FALSE)
range_out_y	A boolean indicating if the output should be a subset element ranges object for element_ranges_y instead of the default dataframe. (default = FALSE)
overlap_threshold	Single numerical input for significant base-pair overlap to be considered 'overlapping'. Default is 1 bp.

Value

Returns a dataframe indicating the unique element links present in the form of their first metadata columns. The element_ranges_x are always under column 1, and element_ranges_y are always under column 2.

Examples

```
# Load enhancer and promoter elements into an ElementRanges object
enhancers <- system.file("extdata/elements", "enhancers.bed", package = "LoopRig", mustWork = TRUE)
promoters <- system.file("extdata/elements", "promoters.bed", package = "LoopRig", mustWork = TRUE)
element_ranges <- ElementsToRanges(enhancers, promoters,
element_names = c("enhancers", "promoters"),
custom_cols = 1, custom_mcols = 4)

# Load loops into LoopRanges object and determine consensus loops (keep all)
ovary_loops <- system.file("extdata/loops", "ovary_hg19.bedpe",
package = "LoopRig", mustWork = TRUE)
pancreas_loops <- system.file("extdata/loops", "pancreas_hg19.bedpe",
package = "LoopRig", mustWork = TRUE)
loops <- LoopsToRanges(ovary_loops, pancreas_loops, custom_cols = 0)
consensus_loops <- ConsensusLoops(loops, keep_all = TRUE)

# Based on consensus loops, determine which enhancers and promoters
# are connected based on loop anchors
LinkedElements(consensus_loops, element_ranges[[1]], element_ranges[[2]])
```

LoopsToRanges *Create a list of ranges objects from looping data*

Description

Uses tab delimited looping data in the form of BEDPE files to create custom GRanges loop objects

Usage

```
LoopsToRanges(..., loop_names = NULL, custom_cols = 0,
  custom_mcols = NULL)
```

Arguments

...	Any number of tab delimited loop data files in BEDPE format
loop_names	A character vector of names for the loop datasets (optional)
custom_cols	An integer indicating the number of extra columns in the BEDPE file (default = 0)
custom_mcols	An integer or vector of integers indicating which columns are used for metadata (optional)

Value

A *LoopRanges* class object: list of GRanges looping data objects

Examples

```
# Load loops into LoopRanges object
ovary_loops <- system.file("extdata/loops", "ovary_hg19.bedpe",
package = "LoopRig", mustWork = TRUE)
spleen_loops <- system.file("extdata/loops", "spleen_hg19.bedpe",
package = "LoopRig", mustWork = TRUE)
LoopsToRanges(ovary_loops, spleen_loops, custom_cols = 0)
```

ScaffoldElements *Finds elements within loops scaffolded by another set of elements*

Description

Using a finalized *LoopRanges* loops object and two sets of elements, determines which loops are 'anchored' by the first set of elements (x) and finds which elements from the second set (y) are within these loops.

Usage

```
ScaffoldElements(loop_ranges, element_ranges_x, element_ranges_y,
  range_out_x = FALSE, range_out_y = FALSE, overlap_threshold = 1)
```

Arguments

loop_ranges	A single *LoopRanges* object that has been subset to one range through ConsensusLoops().
element_ranges_x	The first element ranges to be considered. If using an 'ElementRanges' class object, subset by using list indexing.
element_ranges_y	The second element ranges to be considered. If using an 'ElementRanges' class object, subset by using list indexing.
range_out_x	A boolean indicating if the output should be a subset element ranges object for element_ranges_x instead of the default dataframe. (default = FALSE)
range_out_y	A boolean indicating if the output should be a subset element ranges object for element_ranges_y instead of the default dataframe. (default = FALSE)
overlap_threshold	Single numerical input for significant base-pair overlap to be considered 'overlapping'. Default is 1 base-pair.

Value

Returns a dataframe indicating scaffold linked elements in the form of their first metadata columns and the loop IDs for their scaffolds. The element_ranges_x (scaffold) are always under column 1, and element_ranges_y (links) are always under column 2.

Examples

```
# Load enhancer and promoter elements into an ElementRanges object
enhancers <- system.file("extdata/elements", "enhancers.bed", package = "LoopRig", mustWork = TRUE)
promoters <- system.file("extdata/elements", "promoters.bed", package = "LoopRig", mustWork = TRUE)
element_ranges <- ElementsToRanges(enhancers, promoters,
element_names = c("enhancers", "promoters"),
custom_cols = 1, custom_mcols = 4)

# Load loops into LoopRanges object and determine consensus loops
ovary_loops <- system.file("extdata/loops", "ovary_hg19.bedpe",
package = "LoopRig", mustWork = TRUE)
pancreas_loops <- system.file("extdata/loops", "pancreas_hg19.bedpe",
package = "LoopRig", mustWork = TRUE)
loops <- LoopsToRanges(ovary_loops, pancreas_loops, custom_cols = 0)
consensus_loops <- ConsensusLoops(loops)

# Based on consensus loops, determine which loops are anchored by enhancers and which
# promoters are overlapping with these loops
ScaffoldElements(consensus_loops, element_ranges[[1]], element_ranges[[2]])
```

`StackedElements` *Finds elements stacked atop loop anchors*

Description

Using a finalized *LoopRanges* loops object and two sets of elements, determines which pairs of elements from the two sets are found overlapping with each other and a loop anchor.

Usage

```
StackedElements(loop_ranges, element_ranges_x, element_ranges_y,
  range_out_x = FALSE, range_out_y = FALSE, overlap_threshold = 1)
```

Arguments

`loop_ranges`	A single *LoopRanges* object that has been subset to one range through ConsensusLoops().
`element_ranges_x`	The first element ranges to be considered. If using an 'ElementRanges' class object, subset by using list indexing.
`element_ranges_y`	The second element ranges to be considered. If using an 'ElementRanges' class object, subset by using list indexing.
`range_out_x`	A boolean indicating if the output should be a subset element ranges object for element_ranges_x instead of the default dataframe. (default = FALSE)
`range_out_y`	A boolean indicating if the output should be a subset element ranges object for element_ranges_y instead of the default dataframe. (default = FALSE)
`overlap_threshold`	Single numerical input for significant base-pair overlap to be considered 'overlapping'. Default is 1 base-pair.

Value

Returns a dataframe indicating stacked elements in the form of their first metadata columns. The element_ranges_x are always under column 1, and element_ranges_y are always under column 2.

Examples

```
# Load enhancer and promoter elements into an ElementRanges object
enhancers <- system.file("extdata/elements", "enhancers.bed", package = "LoopRig", mustWork = TRUE)
promoters <- system.file("extdata/elements", "promoters.bed", package = "LoopRig", mustWork = TRUE)
element_ranges <- ElementsToRanges(enhancers, promoters,
element_names = c("enhancers", "promoters"),
custom_cols = 1, custom_mcols = 4)

# Load loops into LoopRanges object and determine consensus loops
ovary_loops <- system.file("extdata/loops", "ovary_hg19.bedpe",
```

```
package = "LoopRig", mustWork = TRUE)
pancreas_loops <- system.file("extdata/loops", "pancreas_hg19.bedpe",
package = "LoopRig", mustWork = TRUE)
loops <- LoopsToRanges(ovary_loops, pancreas_loops, custom_cols = 0)
consensus_loops <- ConsensusLoops(loops)

# Determine pairs of enhancers and promoters stacked on loop anchors
StackedElements(consensus_loops, element_ranges[[1]], element_ranges[[2]])
```

Index