Package ‘plyinteractions’

March 1, 2024

Title Extending tidy verbs to genomic interactions

Description Operate on `GInteractions` objects as tabular data using `dplyr`-like verbs. The functions and methods in `plyinteractions` provide a grammatical approach to manipulate `GInteractions`, to facilitate their integration in genomic analysis workflows.

Version 1.0.0

Date 2023-08-21

License Artistic-2.0

URL https://github.com/js2264/plyinteractions

BugReports https://github.com/js2264/plyinteractions/issues

biocViews Software, Infrastructure

Encoding UTF-8

Roxygen list(markdown = TRUE)

Imports InteractionSet, GenomeInfoDb, BiocGenerics, GenomicRanges, plyranges, IRanges, S4Vectors, rlang, dplyr, tibble, tidyselect, methods, utils

Suggests tidyverse, BSgenome.Mmuseculus.UCSC.mm10, Biostrings, BiocParallel, scales, HiContactsData, rtracklayer, BiocStyle, covr, knitr, rmarkdown, sessioninfo, testthat (>= 3.0.0), RefManageR

Config/testthat/edition 3

VignetteBuilder knitr

RoxygenNote 7.2.3

Collate 'AllGenerics.R' 'AllClasses.R' 'anchor.R' 'annotate.R'
 'arrange.R' 'count-overlaps.R' 'count.R' 'data.R'
 'filter-overlaps.R' 'filter.R' 'find-overlaps.R' 'flank.R'
 'ginteractions-construct.R' 'ginteractions-env.R'
 'ginteractions-getters.R' 'ginteractions-scoping.R'
 'ginteractions-setters.R' 'tbl_vars.R' 'group_data.R'
 'group_by.R' 'internals.R' 'join-overlap-left.R'
 'methods-AnchoredPinnedGInteractions.R'
R topics documented:

'methods-DelegatingGInteractions.R'
'methods-GroupedGInteractions.R'
'methods-PinnedGInteractions.R' 'methods-show.R' 'mutate.R'
'pin.R' 'plyinteractions.R' 'reexports-dplyr.R'
'reexports-plyranges.R' 'reexports.R' 'rename.R'
'replace-anchors.R' 'select.R' 'shift.R' 'slice.R' 'stretch.R'
'summarize.R'

LazyData false

Depends R (>= 4.3.0)

git_url https://git.bioconductor.org/packages/plyinteractions
git_branch RELEASE_3_18
git_last_commit ee766a1
git_last_commit_date 2023-10-24

Repository Bioconductor 3.18

Date/Publication 2024-02-29

Author Jacques Serizay [aut, cre]

Maintainer Jacques Serizay <jacquesserizay@gmail.com>

R topics documented:

plyinteractions-package ........................................... 3
anchors1 .......................................................... 4
annotate ............................................................ 6
as_ginteractions .................................................... 8
ce10_ARCC .......................................................... 10
ce10_REs ............................................................ 11
delegating-ginteractions-methods .................................. 11
dplyr-arrange ......................................................... 13
dplyr-count ........................................................... 14
dplyr-filter ........................................................... 15
dplyr-group_by ......................................................... 16
dplyr-mutate .......................................................... 18
dplyr-rename .......................................................... 20
dplyr-select ........................................................... 21

dplyr-slice ........................................................... 22
dplyr-summarize ......................................................... 23
ginteractions-anchor ............................................... 24
ginteractions-count-overlaps ..................................... 26
ginteractions-filter-overlaps ..................................... 27
ginteractions-find-overlaps ....................................... 29
ginteractions-join-overlap-left .................................. 31
GM12878_HiCCUPS .................................................... 33
group-group_data ..................................................... 34
pin ................................................................. 35
plyranges-flank ....................................................... 37
plyinteractions-package

plyranges-shift ................................................................. 39
plyranges-stretch ............................................................. 41
reexports ................................................................. 42
replace_anchors ............................................................... 43
set_seqnames1 ................................................................. 44

Index 47

plyinteractions-package

plyinteractions: a grammar of data manipulation for genomic interactions

Description

plyinteractions is a dplyr-like API to the GInteractions infrastructure in Bioconductor.

Details

plyinteractions provides a consistent interface for importing and wrangling genomic interactions from a variety of sources. The package defines a grammar of genomic interactions manipulation through a set of verbs. These verbs can be used to construct human-readable analysis pipelines based on GInteractions.

• Group genomic interactions with group_by;
• Summarize grouped genomic interactions with summarize;
• Tally/Count grouped genomic interactions with tally and count;
• Modify genomic interactions with mutate;
• Subset genomic interactions with filter using <data-masking> and logical expressions;
• Pick out any columns from the associated metadata with select using <tidy-select> arguments;
• Subset using indices with slice;
• Order genomic interactions with arrange using categorical/numerical variables.

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

Author(s)

Maintainer: Jacques Serizay <jacquesserizay@gmail.com>

See Also

Useful links:

• https://github.com/js2264/plyinteractions
• Report bugs at https://github.com/js2264/plyinteractions/issues
Enhanced GInteractions getters

Description
Enhanced GInteractions getters

Usage
anchors1(x)
anchors2(x)
seqnames1(x)
seqnames2(x)
start1(x)
start2(x)
end1(x)
end2(x)
width1(x)
width2(x)
strand1(x)
strand2(x)
ranges1(x)
ranges2(x)

## S4 method for signature 'GInteractions'
x$name

## S4 method for signature 'GInteractions'
anchors1(x)

## S4 method for signature 'GInteractions'
anchors2(x)

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

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

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

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

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

## S4 method for signature 'GInteractions'
end2(x)

## S4 method for signature 'GInteractions'
width1(x)

## S4 method for signature 'GInteractions'
width2(x)

## S4 method for signature 'GInteractions'
strand1(x)

## S4 method for signature 'GInteractions'
strand2(x)

## S4 method for signature 'GInteractions'
ranges1(x)

## S4 method for signature 'GInteractions'
ranges2(x)

Arguments

x a GInteractions object

name The pattern or name of a column stored in the GInteractions metadata (mcols).

Value

One of the core GInteractions fields (e.g. seqnames1, start1, ...) or one of the metadata columns when using $. Note that auto-completion works with $.

Examples

gi <- data.frame(
  seqnames1 = 'chr1', start1 = 1, end1 = 10,
annotate

Annotate both anchors of a GInteractions

Description

For each interaction in a GInteractions object, annotate returns the pairs of annotations from the GRanges object it overlaps with.

Usage

annotate(x, y, by)
annotate_directed(x, y, by)

## S4 method for signature 'GInteractions,GRanges,character'
annotate(x, y, by)
annotate_directed(x, y, by)

## S4 method for signature 'GInteractions,GRanges,character'
annotate(x, y, by)
annotate_directed(x, y, by)

Arguments

x a GInteractions object
y a GRanges object to extract annotations from
by Column name from y to use to extract annotations

Value

a GInteractions object with two extra metadata columns named by.1 and by.2.
Examples

####################################################################
# 1. Basic example
####################################################################

```r
gi <- read.table(text = "
  chr1 11 20 - chr1 21 30 +
  chr1 21 30 + chr2 51 60 +",
col.names = c("seqnames1", "start1", "end1", "strand1",
  "seqnames2", "start2", "end2", "strand2")
)
gi |> as_ginteractions()

gr <- GenomicRanges::GRanges(c("chr1:20-30:+", "chr2:55-65:+")) |
  plyranges::mutate(id = 1:2)

annotate(gi, gr, by = 'id')
annotate_directed(gi, gr, by = 'id')
```

####################################################################
# 2. Match loops with tiled genomic bins
####################################################################

data(GM12878_HiCCUPS)
loops <- GM12878_HiCCUPS |
  pin_by('first') |
  anchor_center() |
  mutate(width1 = 500) |
  pin_by('second') |
  anchor_center() |
  mutate(width2 = 500)

genomic_bins <- GenomeInfoDb::getChromInfoFromUCSC(
  'hg19', assembled.molecules.only = TRUE, as.Seqinfo = TRUE
) |
  GenomicRanges::tileGenome(tilewidth = 10000) |
  unlist() |
  plyranges::mutate(binID = seq_len(plyranges::n()))

annotate(loops, genomic_bins, by = 'binID') |
  select(starts_with('binID'))

```

####################################################################
# 3. Annotate interactions by a set of regulatory elements
####################################################################

data(ce10_ARCC)
data(ce10_REs)
annotate(ce10_ARCC, ce10_REs, by = 'annot') |
  count(annot.1, annot.2) |>
```
as.data.frame() |>
dplyr::arrange(desc(n))

---

**as_ginteractions**  
*Construct a GInteractions object from a tibble, DataFrame or data.frame*

**Description**

The `as_ginteractions` function looks for column names in `.data` called `seqnames{1,2}`, `start{1,2}`, `end{1,2}`, and `strand{1,2}` in order to construct a GInteractions object. By default other columns in `.data` are placed into the `mcols` (metadata columns) slot of the returned object.

**Usage**

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

**Arguments**

- `.data`  
  A `data.frame`, `DataFrame` or tibble() to construct a GInteractions object from.

- `...`  
  Optional named arguments specifying which the columns in `.data` contain in the core components a GInteractions object.

- `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.

- `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.

**Value**

a GInteractions object.

**See Also**

`InteractionSet::GInteractions()`
as_ginteractions

Examples

# 1. GInteractions from bedpe files imported into a data.frame

```r
bedpe <- read.table(text = "
chr1 100 200 chr1 5000 5100 bedpe_example1 30 + -
chr1 1000 5000 chr1 3000 3800 bedpe_example2 100 + -",
col.names = c(
  "chrom1", "start1", "end1",
  "chrom2", "start2", "end2", "name", "score", "strand1", "strand2")
)
```

```r
table <- read.table(text = "
chr1 100 200 chr1 5000 5100
chr1 1000 5000 chr1 3000 3800",
col.names = c("chr1", "start1", "end1")
)
```

```r
as_ginteractions(seqnames1 = chrom1, seqnames2 = chrom2)
```

# 2. GInteractions from standard pairs files imported into a data.frame

```r
pairs <- read.table(text = "
pair1 chr1 10000 chr1 20000 + +
pair2 chr1 50000 chr1 70000 + +
pair3 chr1 60000 chr2 10000 + +
pair4 chr1 30000 chr3 40000 + -",
col.names = c(
  "pairID", "chr1", "pos1", "chr2", "pos2", "strand1", "strand2")
)
```

```r
as_ginteractions(
  seqnames1 = chr1, start1 = pos1, width1 = 1000,
  seqnames2 = chr2, start2 = pos2, width2 = 1000,
  starts.in.df.are.0based = TRUE
)
```

# 3. GInteractions from data.frame with extra fields

```r
df <- read.table(text = "
chr1 100 200 chr1 5000 5100
chr1 1000 5000 chr1 3000 3800",
col.names = c("chr1", "start1", "end1", "chr2", "start2", "end2")
)
```

```r
as_ginteractions(seqnames1 = chr1, seqnames2 = chr2)
```

```r
df <- read.table(text = "
chr1 100 200 chr1 5000 5100
chr1 1000 5000 chr1 3000 3800",
col.names = c("chr1", "start1", "end1", "chr2", "start2", "end2")
)
```
df |>  
  as_ginteractions(  
    seqnames1 = chr1, seqnames2 = chr2, strand1 = '+', strand2 = '-'  
  )

data.frame(type = "cis", count = 3) |>  
  as_ginteractions(  
    seqnames1 = 'chr1', start1 = 1, end1 = 10,  
    seqnames2 = 'chr1', start2 = 40, end2 = 50  
  )

# 4. GInteractions from a real like pairs files
pairsf <- system.file('extdata', 'pairs.gz', package = 'plyinteractions')
pairs <- read.table(pairsf, comment.char = '#', header = FALSE)
head(pairs)
pairs |>  
  as_ginteractions(  
    seqnames1 = V2, start1 = V3, width1 = 1, strand1 = V6,  
    seqnames2 = V4, start2 = V5, width2 = 1, strand2 = V7,  
    starts.in.df.are.0based = TRUE  
  )

ce10_ARCC  

Interactions identified in L3 C. elegans by ARC-C

Description
Supplemental Table 2 obtained from Genome Biology online publication.
Huang N, Seow WQ, Appert A, Dong Y, Stempor P and Ahringer J Accessible Region Conformation Capture (ARC-C) gives high-resolution insights into genome architecture and regulation. Genome Res 2022 Feb;32(2):357-366. PMID: 34933938

Usage
ce10_ARCC

Format
An object of class GInteractions of length 14992.

Value
A GInteractions object

Source
https://genome.cshlp.org/content/early/2021/12/21/gr.275669.121
Description
Figure 2 - Source data 1 obtained from eLife online publication.

Usage
ce10_REs

Format
An object of class GRanges of length 42245.

Value
A GRanges object

Source
https://genome.cshlp.org/content/early/2021/12/21/gr.275669.121

deleating-ginteractions-methods

Methods for Delegating GInteractions objects

Description
Methods for DelegatingGIInteractions objects

Usage
## S4 method for signature 'DelegatingGIInteractions'
anchors1(x)

## S4 method for signature 'DelegatingGIInteractions'
ranges1(x)

## S4 method for signature 'DelegatingGIInteractions'
seqnames1(x)
## S4 method for signature 'DelegatingGInteractions'

start1(x)

## S4 method for signature 'DelegatingGInteractions'

end1(x)

## S4 method for signature 'DelegatingGInteractions'

width1(x)

## S4 method for signature 'DelegatingGInteractions'

strand1(x)

## S4 method for signature 'DelegatingGInteractions'

anchors2(x)

## S4 method for signature 'DelegatingGInteractions'

ranges2(x)

## S4 method for signature 'DelegatingGInteractions'

seqnames2(x)

## S4 method for signature 'DelegatingGInteractions'

start2(x)

## S4 method for signature 'DelegatingGInteractions'

end2(x)

## S4 method for signature 'DelegatingGInteractions'

width2(x)

## S4 method for signature 'DelegatingGInteractions'

strand2(x)

## S4 method for signature 'DelegatingGInteractions'

anchors(x)

## S4 method for signature 'DelegatingGInteractions'

regions(x)

## S4 method for signature 'DelegatingGInteractions'

seqinfo(x)

## S4 method for signature 'DelegatingGInteractions'

mcols(x)

## S4 method for signature 'DelegatingGInteractions'

show(object)
**Value**

One of the core GInteractions fields (e.g. seqnames1, start1, ...)

---

**Description**

Arrange a GInteractions by a column

**Usage**

```
## S3 method for class 'GInteractions'
arrange(.data, ...)
```

**Arguments**

- `.data` a GInteractions object
- `...` Variables, or functions of variables. Use dplyr::desc() to sort a variable in descending order.

**Value**

a GInteractions object.

**Examples**

```r
gi <- read.table(text = "
chr1 1 10 chr1 1 10
chr1 2 10 chr2 1 10
chr3 3 10 chr3 1 10
chr4 4 10 chr4 1 10
chr5 5 10 chr5 1 10",
col.names = c("seqnames1", "start1", "end1",
"seqnames2", "start2", "end2")
) |> as_ginteractions() |> mutate(cis = seqnames1 == seqnames2, score = runif(5)*100, gc = runif(5))
gi
```

```
# 1. Arrange GInteractions by a numerical column

gi |> arrange(gc)
```

```
# 2. Arrange GInteractions by a logical column
```

gi |> arrange(cis)

# 3. Arrange GInteractions by a factor

gi |> mutate(rep = factor(c("rep1", "rep2", "rep1", "rep2", "rep1"))) |> arrange(rep)

# 4. Combine sorting variables

gi |> mutate(rep = factor(c("rep1", "rep2", "rep1", "rep2", "rep1"))) |> arrange(dplyr::desc(rep), score)

dplyr-count

Count or tally GInteractions per group

Description

Count or tally GInteractions per group

Usage

## S3 method for class 'GroupedGInteractions'
tally(x, wt = NULL, sort = FALSE, name = NULL)

## S3 method for class 'GroupedGInteractions'
count(x, ..., wt = NULL, sort = FALSE, name = NULL)

## S3 method for class 'GInteractions'
count(x, ..., wt = NULL, sort = FALSE, name = NULL)

Arguments

x  A grouped GInteractions object

wt  <data-masking> Frequency weights. Can be NULL or a variable:
    • If NULL (the default), counts the number of rows in each group.
    • If a variable, computes sum(wt) for each group.

sort  If TRUE, will show the largest groups at the top.

name  The name of the new column in the output.

...  <data-masking> Variables to group by.
Value

A S4Vectors::DataFrame() object, with an added column with the count/tably per group.

Examples

gi <- read.table(text = "
chr1 11 20 chr1 21 30 + +
chr1 11 20 chr1 51 55 + +
chr1 11 30 chr1 51 55 - -
chr1 11 30 chr2 51 60 - -",
col.names = c("seqnames1", "start1", "end1",
"seqnames2", "start2", "end2", "strand1", "strand2")
) |> as_ginteractions() |> mutate(score = runif(4), type = c('cis', 'cis', 'cis', 'trans'))

# 1. Tally groups

gi

gi |> group_by(strand1) |> tally()

gi |> group_by(type) |> tally()

gi |> group_by(type) |> tally(wt = score)

# 2. Count per groups

gi |> count(type)

gi |> group_by(type) |> count(strand1)

# dplyr-filter

Subset a GInteractions with tidyverse-like filter

Description

Subset a GInteractions with tidyverse-like filter

Usage

## S3 method for class 'GInteractions'
filter(.data, ...)
Arguments

.data a GInteractions object

Expressions that return a logical value, and are defined in terms of the variables in .data. If multiple expressions are included, they are combined with the & operator. Only rows for which all conditions evaluate to TRUE are kept.

Value

a GInteractions object.

Examples

```r
gi <- read.table(text = "chr1 1 10 chr1 1 10
chr1 2 10 chr2 1 10
chr3 3 10 chr3 1 10
chr4 4 10 chr4 1 10
chr5 5 10 chr5 1 10",
col.names = c("seqnames1", "start1", "end1",
"seqnames2", "start2", "end2")
) |> as_ginteractions() |> mutate(cis = seqnames1 == seqnames2, score = runif(5)*100, gc = runif(5))
gi
```

# 1. Filter metadata columns from GInteractions by condition

```r
gi |> filter(gc > 0.1)
gi |> filter(gc > 0.1, score > 50)
gi |> filter(cis)
```

# 2. On-the-fly calculations

```r
gi
```

```r
gi |> filter(start1 >= start2 + 3)
gi |> filter(score * gc > score * 0.5)
```

Description

Group GInteractions by columns
Usage

## S3 method for class 'GInteractions'
group_by(.data, ..., .add = FALSE)

## S3 method for class 'DelegatingGInteractions'
group_by(.data, ..., .add = FALSE)

## S3 method for class 'GroupedGInteractions'
ungroup(x, ...)

Arguments

.data, x  a (Grouped)GInteractions object
...  Column(s) to group by.
.add  When FALSE, the default, group_by() will override existing groups. To add to
the existing groups, use .add = TRUE.

Value

a GroupedGInteractions object. When a (Anchored)PinnedGInteractions object is grouped,
both anchoring and pinning are dropped.

Examples

```r
gi <- read.table(text =
  chr1 11 20 chr1 21 30
  chr1 11 20 chr1 51 55
  chr1 11 30 chr1 51 55
  chr1 11 30 chr2 51 60",
col.names = c(  
  "seqnames1", "start1", "end1",  
  "seqnames2", "start2", "end2"
  ) |>
  as_ginteractions() |>
  mutate(type = c('cis', 'cis', 'cis', 'trans'), score = runif(4))

# 1. Group by core column

gi |> group_by(end1)

gi |> group_by(end1, end2) |> group_data()

# 2. Group by metadata column

gi |> group_by(type) |> group_data()
```


### 3. Combine core and metadata column grouping

```r
gi |> group_by(end1, type)
gi |> group_by(end1, type) |> group_data()
```

### 4. Create a new column and group by this new variable

```r
gi |> group_by(class = c(1, 2, 1, 2))
```

### 5. Replace or add groups to a GroupedGInteractions

```r
ggi <- gi |> group_by(class = c(1, 2, 1, 2))
ggi |> group_data()
ggi |> group_by(type) |> group_data()
ggi |> group_by(type, .add = TRUE) |> group_data()
```

### 6. Ungroup GInteractions

```r
ggi <- gi |> group_by(type, class = c(1, 2, 1, 2))
ggi
ungroup(ggi, type)
ungroup(ggi, class)
```

---

dplyr-mutate  

**Mutate columns from a GInteractions object**

### Description

Mutate columns from a GInteractions object

### Usage

```r
## S3 method for class 'GInteractions'
mutate(.data, ...)
```

### Arguments

- `.data` a GInteractions object
- `...` Optional named arguments specifying which the columns in `.data` to create/modify.
Value

a GInteractions object.

Examples

gi <- read.table(text = "
chr1 10 20 chr1 50 51
chr1 10 50 chr2 30 40",
col.names = c("chr1", "start1", "end1", "chr2", "start2", "end2")) |>
as_ginteractions(seqnames1 = chr1, seqnames2 = chr2)

# 1. Add metadata columns to a GInteractions object

gi |>
mutate(type = c('cis', 'trans'), score = runif(2)) |>
mutate(type2 = type)

# 2. More complex, nested or inplace changes

gi |>
mutate(type = c('cis', 'trans'), score = runif(2)) |>
mutate(type2 = type) |>
mutate(count = c(1, 2), score = count * 2, new_col = paste0(type2, score))

# 3. Core GInteractions columns can also be modified

gi |>
mutate(start1 = 1, end1 = 10, width2 = 30, strand2 = c('-', '+'))

# Note how the core columns are modified sequentially

gi |>
mutate(start1 = 1, end1 = 10)

gi |>
mutate(start1 = 1, end1 = 10, width1 = 50)

# 4. Evaluating core GInteractions columns

gi |>
mutate(score = runif(2),
cis = seqnames1 == seqnames2,
distance = ifelse(cis, start2 - end1, NA)
**dplyr-rename**

Renaming columns from a `GInteractions` object with tidyverse-like syntax.

### Description

Rename columns from a `GInteractions` with tidyverse-like `rename`.

### Usage

```r
## S3 method for class 'GInteractions'
rename(.data, ...)  
```

### Arguments

- `.data`: a `GInteractions` object
- `...`: Use `new_name = old_name` to rename selected variables.

### Value

A `GInteractions` object.

### Examples

```r
gi <- read.table(text = |
chr1 10 20 chr1 50 51  
chr1 10 50 chr2 30 40|, |
col.names = c("chr1", "start1", "end1", "chr2", "start2", "end2")) |>
  as_ginteractions(seqnames1 = chr1, seqnames2 = chr2) |>
  mutate(type = c('cis', 'trans'), score = runif(2))

# 1. Rename metadata columns to a GInteractions object

```
Select columns within GInteractions metadata columns

Usage

## S3 method for class 'GInteractions'
select(.data, ..., .drop_ranges = FALSE)

Arguments

- `.data` a GInteractions object
- `...` Integer indicating rows to keep.
- `.drop_ranges` if TRUE, returns a DataFrame object. In this case, it enables selection of any column including core GInteractions columns.

Value

a GInteractions object.

Examples

```r
gi <- read.table(text = 
  "chr1 1 10 chr1 1 10
  chr2 1 10 chr2 1 10
  chr3 1 10 chr3 1 10
  chr4 1 10 chr4 1 10
  chr5 1 10 chr5 1 10",
col.names = c(
  "seqnames1", "start1", "end1",
  "seqnames2", "start2", "end2")
) |> as_ginteractions() |> mutate(score = runif(5)*100, cis = TRUE, gc = runif(5))
```
gi |> select(gc, score)

# 3. Select metadata columns from GInteractions with <tidy-select>

gi |> select(contains('s'))

# 4. Select core and metadata columns with .drop_ranges = TRUE

gi |> select(matches('^s'), .drop_ranges = TRUE)

dplyr-slice

Slice a GInteractions rows by their index

Description
Slice a GInteractions rows by their index

Usage
## S3 method for class 'GInteractions'
slice(.data, ...)

Arguments
.data a GInteractions object
...

Value
a GInteractions object.

Examples

```r
gi <- read.table(text = "
chr1  1  10 chr1  1  10
chr2  1  10 chr2  1  10
chr3  1  10 chr3  1  10
chr4  1  10 chr4  1  10
chr5  1  10 chr5  1  10",
col.names = c("seqnames1", "start1", "end1",
              "seqnames2", "start2", "end2")
) |> slice
```
# Slice a GInteractions

```r
gi |> slice(1, 2, 3)
gi |> slice(-3)
gi |> slice(1:2, 5:4)
```

---

**Description**

Summarize GInteractions per group

**Usage**

```r
## S3 method for class 'GroupedGInteractions'
summarise(.data, ...)

## S3 method for class 'GroupedGInteractions'
summarize(.data, ...)
```

**Arguments**

- `.data` a (grouped) GInteractions object
- `...` Name-value pairs of summary functions. The name will be the name of the variable in the result.

**Value**

A `S4Vectors::DataFrame()` object:

- The rows come from the underlying `group_keys()`.
- The columns are a combination of the grouping keys and the summary expressions that you provide.
- GInteractions class is **not** preserved, as a call to `summarize` fundamentally creates a new data frame

**Examples**

```r
gi <- read.table(text = "
chr1 11 30 chr1 21 30 + +
chr1 11 30 chr1 51 55 + +
chr1 11 30 chr1 51 55 - -
chr1 11 30 chr2 51 60 - -",
```

---

**dplyr-summarize**

Summarize GInteractions per group
col.names = c("seqnames1", "start1", "end1", "seqnames2", "start2", "end2", "strand1", "strand2") |>
  as_ginteractions() |>
  mutate(score = runif(4), type = c('cis', 'cis', 'cis', 'trans'))

# 1. Summarize a single column

```r
gi

gi |> group_by(type) |> summarize(m = mean(score))

# 2. Summarize by multiple columns

```r
gi |>
  group_by(strand1, seqnames2) |>
  summarise(m = mean(score), n = table(type))
```

---

**ginteractions-anchor**

Manage GInteractions anchors with plyranges

**Description**

Manage GInteractions anchors with plyranges

**Usage**

```r
## S3 method for class 'AnchoredPinnedGInteractions'
anchor(x)

## S3 method for class 'AnchoredPinnedGInteractions'
unanchor(x)

## S3 method for class 'PinnedGInteractions'
anchor_start(x)
```
Arguments

x  A PinnedGIInteractions object

Value

- anchor_* functions return an AnchoredPinnedGIInteractions object.
- anchor returns a character string indicating where the pinned anchors are anchored at.
- unanchor removes the anchoring for a AnchoredPinnedGIInteractions object.

Examples

```r
gi <- read.table(text = "chr1 11 20 chr1 21 30 + +
chr1 11 20 chr1 51 55 + +
chr1 11 30 chr1 51 55 - -
chr1 11 30 chr2 51 60 - -",
col.names = c("seqnames1", "start1", "end1",
"seqnames2", "start2", "end2", "strand1", "strand2")
)
```

```r
as_ginteractions() |>
mutate(score = runif(4), type = c('cis', 'cis', 'cis', 'trans'))

####################################################################
# 1. Anchoring pinned genomic interactions with plyranges
####################################################################

gi |> pin_by("second") |> anchor_end()

ginteractions-count-overlaps

*Count overlaps between a query GInteractions and a GRanges*

### Description

Count overlaps between a query GInteractions and a GRanges

### Usage

```r
## S3 method for class 'PinnedGInteractions'
count_overlaps(x, y, maxgap = -1L, minoverlap = 0L)

## S3 method for class 'GInteractions'
count_overlaps(x, y, maxgap = -1L, minoverlap = 0L)

## S3 method for class 'PinnedGInteractions'
count_overlaps_directed(x, y, maxgap = -1L, minoverlap = 0L)

## S3 method for class 'GInteractions'
count_overlaps_directed(x, y, maxgap = -1L, minoverlap = 0L)
```

### Arguments

- **x**: A (Pinned)GInteractions object
- **y**: A GRanges object
- **maxgap**, **minoverlap**: See ?countOverlaps in the GenomicRanges package for a description of these arguments

### Value

An integer vector of same length as x.

**Pinned GInteractions**

When using count_overlaps() with a PinnedGInteractions object, only the pinned anchors are used to check for overlap with y. This is equivalent to specifying use.region="both" in InteractionSet::findOverlaps().
Examples

```r
gi <- read.table(text = "
  chr1 11 20 - chr1 21 30 +
  chr1 11 20 - chr1 51 55 +
  chr1 21 30 - chr1 51 55 +
  chr1 21 30 - chr2 51 60 +",
  col.names = c("seqnames1", "start1", "end1", "strand1",
               "seqnames2", "start2", "end2", "strand2")
) |> as_ginteractions() |> mutate(id = 1:4, type = 'gi')

gr <- GenomicRanges::GRanges(
  c("chr1:20-30:+", "chr2:55-65:-")
) |> plyranges::mutate(id = 1:2, type = 'gr')

gi

gr

# 1. Count overlaps between GInteractions and a subject GRanges

count_overlaps(gi, gr)

count_overlaps_directed(gi, gr)

# 2. Count overlaps between PinnedGInteractions and a subject GRanges

gi |> pin_by("first") |> count_overlaps(gr)

gi |> pin_by("second") |> count_overlaps(gr)

gi |> pin_by("first") |> count_overlaps_directed(gr)

gi |> pin_by("second") |> count_overlaps_directed(gr)
```

---

**ginteractions-filter-overlaps**

Filter GInteractions overlapping with a GRanges

---

**Description**

Filter GInteractions overlapping with a GRanges
Usage

```r
## S3 method for class 'PinnedGInteractions'
filter_by_overlaps(x, y, maxgap = -1L, minoverlap = 0L)

## S3 method for class 'GInteractions'
filter_by_overlaps(x, y, maxgap = -1L, minoverlap = 0L)

## S3 method for class 'PinnedGInteractions'
filter_by_non_overlaps(x, y, maxgap = -1L, minoverlap = 0L)

## S3 method for class 'GInteractions'
filter_by_non_overlaps(x, y, maxgap = -1L, minoverlap = 0L)
```

Arguments

- `x`: A (Pinned)GInteractions object
- `y`: A GRanges object
- `maxgap`, `minoverlap`:
  See ?countOverlaps in the GenomicRanges package for a description of these arguments

Value

An integer vector of same length as `x`.

Pinned GInteractions

When using `filter_by_overlaps()` with a PinnedGInteractions object, only the pinned anchors are used to check for overlap with `y`. This is equivalent to specifying use.region="both" in InteractionSet::findOverlaps().

Examples

```r
gi <- read.table(text = "
chr1 11 20 - chr1 21 30 +
chr1 11 20 - chr1 51 55 +
chr1 21 30 - chr1 51 55 +
chr1 21 30 - chr2 51 60 +",
col.names = c(
  "seqnames1", "start1", "end1", "strand1",
  "seqnames2", "start2", "end2", "strand2")
) |> as_ginteractions() |> mutate(id = 1:4, type = 'gi')

gr <- GenomicRanges::GRanges(
c("chr1:20-30:+", "chr2:55-65:-")
) |> plyranges::mutate(id = 1:2, type = 'gr')

gi
gr
# 1. Filter GInteractions overlapping with a subject GRanges

```r
filter_by_overlaps(gi, gr)

filter_by_non_overlaps(gi, gr)
```

# 2. Filter PinnedGInteractions overlapping with a subject GRanges

```r
gi |> pin_by("first") |> filter_by_overlaps(gr)

gi |> pin_by("first") |> filter_by_non_overlaps(gr)

gi |> pin_by("second") |> filter_by_overlaps(gr)

gi |> pin_by("second") |> filter_by_non_overlaps(gr)
```

---

Find overlaps between a query GInteractions and a GRanges

**Description**

Find overlaps between a query GInteractions and a GRanges

**Usage**

```r
## S3 method for class 'PinnedGInteractions'
find_overlaps(x, y, maxgap = -1L, minoverlap = 0L, suffix = c(".x", ".y"))

## S3 method for class 'GInteractions'
find_overlaps(x, y, maxgap = -1L, minoverlap = 0L, suffix = c(".x", ".y"))

## S3 method for class 'PinnedGInteractions'
find_overlaps_directed(
  x,
  y,
  maxgap = -1L,
  minoverlap = 0L,
  suffix = c(".x", ".y")
)

## S3 method for class 'GInteractions'
find_overlaps_directed(

```
Arguments

\[ x, \]
\[ y, \]
\[ \text{maxgap} = -1L, \]
\[ \text{minoverlap} = 0L, \]
\[ \text{suffix} = \text{c(".x", ".y")} \]
)

Value

A GInteractions object with rows corresponding to the GInteractions in \( x \) that overlap \( y \).

Rationale

Find_overlaps() will search for any overlap between GInteractions in \( x \) and GRanges in \( y \). It will return a GInteractions object of length equal to the number of times \( x \) overlaps \( y \). This GInteractions will have additional metadata columns corresponding to the metadata from \( y \). Find_overlaps_directed() takes the strandness of each object into account.

Pinned GInteractions

When using find_overlaps() with a PinnedGInteractions object, only the pinned anchors are used to check for overlap with \( y \). This is equivalent to specifying use.region="both" in InteractionSet::findOverlaps().

Examples

```r
gi <- read.table(text = "
  chr1 11 20 - chr1 21 30 +
  chr1 11 20 - chr1 51 55 +
  chr1 21 30 - chr1 51 55 +
  chr1 21 30 - chr2 51 60 +",
  col.names = c(
    "seqnames1", "start1", "end1", "strand1",
    "seqnames2", "start2", "end2", "strand2"
  )
)
| as_ginteractions() |> mutate(id = 1:4, type = 'gi')

gr <- GenomicRanges::GRanges(
  c("chr1:20-30:+", "chr2:55-65:-")
)
| plyranges::mutate(id = 1:2, type = 'gr')
```
gi
g

# 1. Find overlaps between GInteractions and a subject GRanges
find_overlaps(gi, gr)
find_overlaps_directed(gi, gr)

# 2. Find overlaps between PinnedGInteractions and a subject GRanges
gi |> pin_by("first") |> find_overlaps(gr)
gi |> pin_by("second") |> find_overlaps(gr)
gi |> pin_by("first") |> find_overlaps_directed(gr)
gi |> pin_by("second") |> find_overlaps_directed(gr)

description
Join overlaps between a query GInteractions and a GRanges

Usage

## S3 method for class 'PinnedGInteractions'
join_overlap_left(x, y, maxgap = -1L, minoverlap = 0L, suffix = c(".x", ".y"))

## S3 method for class 'GInteractions'
join_overlap_left(x, y, maxgap = -1L, minoverlap = 0L, suffix = c(".x", ".y"))

## S3 method for class 'PinnedGInteractions'
join_overlap_left_directed(
x,
y,
maxgap = -1L,
minoverlap = 0L,
suffix = c(".x", ".y")
)
## S3 method for class 'GInteractions'
join_overlap_left_directed(
  x,  
  y,  
  maxgap = -1L,  
  minoverlap = 0L,  
  suffix = c(".x", ".y")
)

### Arguments

- **x**: A (Pinned)GInteractions object
- **y**: A GRanges object
- **maxgap**, **minoverlap**: See ?countOverlaps in the GenomicRanges package for a description of these arguments
- **suffix**: Suffix to add to metadata columns (character vector of length 2, default to c(".x", ".y")).

### Value

An integer vector of same length as x.

### Examples

```r
# example

gi <- read.table(text = "
chr1 11 20 - chr1 21 30 +
chr1 11 20 - chr1 51 55 +
chr1 21 30 - chr1 51 55 +
chr1 21 30 - chr2 51 60 +",
col.names = c("seqnames1", "start1", "end1", "strand1", "seqnames2", "start2", "end2", "strand2")) |> as_ginteractions() |> mutate(id = 1:4, type = 'gi')

gr <- GenomicRanges::GRanges(c("chr1:20-30:+", "chr2:55-65:-")) |> plyranges::mutate(id = 1:2, type = 'gr')

# 1. Join overlaps between GInteractions and a subject GRanges
join_overlap_left(gi, gr)
```
GM12878_HiCCUPS

join_overlap_left_directed(gi, gr)

#########################################################################
# 2. Join overlaps between PinnedGInteractions and a subject GRanges
#########################################################################

gi |> pin_by("first") |> join_overlap_left(gr)

* gi |> pin_by("first") |> join_overlap_left_directed(gr)

* gi |> pin_by("second") |> join_overlap_left(gr)

* gi |> pin_by("second") |> join_overlap_left_directed(gr)

---

GM12878_HiCCUPS  Loops identified in GM12878 with HiCCUPS

Description

File obtained from GEO entry GSE63525 (GSE63525_GM12878_primary+replicate_HiCCUPS_looplist.txt.gz).


Usage

GM12878_HiCCUPS

Format

An object of class GInteractions of length 9448.

Value

A GInteractions object

Source

Description

GInteractions grouping metadata

Usage

```r
## S3 method for class 'GroupedGInteractions'
group_data(.data)

## S3 method for class 'GroupedGInteractions'
group_keys(.tbl, ...)

## S3 method for class 'GroupedGInteractions'
group_indices(.data, ...)

## S3 method for class 'GInteractions'
group_vars(x)

## S3 method for class 'GroupedGInteractions'
group_vars(.data)

## S3 method for class 'GroupedGInteractions'
groups(x)

## S3 method for class 'GroupedGInteractions'
group_size(x)

## S3 method for class 'GroupedGInteractions'
n_groups(x)
```

Arguments

- `.data`, `.tbl`, `x`  a GInteractions object
- `...`  Ignored.

Value

- a GInteractions object.

Examples

```r
gi <- read.table(text = "
chr1 11 20 chr1 21 30
chr1 11 20 chr1 51 55
"
```

chr1 11 30 chr1 51 55
chr1 11 30 chr2 51 60",
col_names = c(
    "seqnames1", "start1", "end1",
    "seqnames2", "start2", "end2"
) |> as_ginteractions() |> mutate(type = c('cis', 'cis', 'cis', 'trans'), score = runif(4))

```r
ggi <- gi |> group_by(end1)
ggi
group_data(ggi)
group_keys(ggi)
group_rows(ggi)
group_indices(ggi)
group_vars(ggi)
groups(ggi)
group_size(ggi)
n_groups(ggi)
```

---

**pin**

*Pin GInteractions by anchors set (anchors1 or anchors2).*

**Description**

Pin GInteractions by anchors set (anchors1 or anchors2).

**Usage**

```r
pin(x, anchors)
```

```r
pin_by(x, anchors)
```

```r
pinned_anchors(x)
```

```r
unpin(x)
```

```r
## S4 method for signature 'GroupedGInteractions,character'
pin(x, anchors)
```

```r
## S4 method for signature 'GroupedGInteractions,numeric'
pin(x, anchors)
```

```r
## S4 method for signature 'GInteractions,character'
pin(x, anchors)
```

```r
## S4 method for signature 'GInteractions,numeric'
pin(x, anchors)
```
### S4 method for signature 'PinnedGInteractions,missing'

`pin(x, anchors)`

### S4 method for signature 'PinnedGInteractions,character'

`pin(x, anchors)`

### S4 method for signature 'PinnedGInteractions,numeric'

`pin(x, anchors)`

### S4 method for signature 'AnchoredPinnedGInteractions,character'

`pin(x, anchors)`

### S4 method for signature 'AnchoredPinnedGInteractions,numeric'

`pin(x, anchors)`

`pin_first(x)`

`pin_second(x)`

`pin_anchors1(x)`

`pin_anchors2(x)`

### S4 method for signature 'AnchoredPinnedGInteractions'

`unpin(x)`

### S4 method for signature 'PinnedGInteractions'

`unpin(x)`

### S4 method for signature 'GInteractions'

`unpin(x)`

### S4 method for signature 'PinnedGInteractions'

`pinned_anchors(x)`

### S4 method for signature 'AnchoredPinnedGInteractions'

`pinned_anchors(x)`

---

**Arguments**

- `x` a GInteractions object
- `anchors` Anchors to pin on ("first" or "second")

**Value**

- `pin_*` functions return a PinnedGInteractions object.
- `pin` returns a numerical value indicating which set of anchors is pinned.
• unpin removes the pinning of a PinnedGInteractions object.
• pinned_anchors returns an (Anchored)GenomicRanges object corresponding to the pinned anchors of a PinnedGInteractions object.

Examples

```r
# Generate flanking regions from pinned anchors of a GInteractions object with plyranges

### Description
Generate flanking regions from pinned anchors of a GInteractions object with plyranges
```
Usage

flank_downstream(x, width)

## S3 method for class 'Ranges'
flank_downstream(x, width)

## S3 method for class 'PinnedGInteractions'
flank_downstream(x, width)

flank_upstream(x, width)

## S3 method for class 'Ranges'
flank_upstream(x, width)

## S3 method for class 'PinnedGInteractions'
flank_upstream(x, width)

flank_right(x, width)

## S3 method for class 'Ranges'
flank_right(x, width)

## S3 method for class 'PinnedGInteractions'
flank_right(x, width)

flank_left(x, width)

## S3 method for class 'Ranges'
flank_left(x, width)

## S3 method for class 'PinnedGInteractions'
flank_left(x, width)

Arguments

x a PinnedGInteractions object

width The width of the flanking region relative to the ranges in x. Either an integer vector of length 1 or an integer vector the same length as x. The width can be negative in which case the flanking region is reversed.

Value

A PinnedGInteractions object

Examples

gi <- read.table(text = "
chr1 11 20 chr1 21 30 + +
)
plyranges-shift

### Description

Shift pinned anchors of a GInteractions object with plyranges

### Usage

```
shift_downstream(x, shift)
```

## S3 method for class 'Ranges'
shift_downstream(x, shift)

## S3 method for class 'PinnedGInteractions'
shift_downstream(x, shift)

```
shift_upstream(x, shift)
```

## S3 method for class 'Ranges'
shift_upstream(x, shift)

## S3 method for class 'PinnedGInteractions'
shift_upstream(x, shift)

---

```
chr1 11 20 chr1 51 55 + +chr1 11 30 chr1 51 55 - -chr1 11 30 chr2 51 60 - -
```

```r
col.names = c("seqnames1", "start1", "end1",
              "seqnames2", "start2", "end2", "strand1", "strand2")
)

as_ginteractions() |

mutate(score = runif(4), type = c("cis", "cis", "cis", "trans"))
```

---

### Simple flanking

```r
gi

gi |> pin_by("first") |> flank_left(-2)

gi |> pin_by("second") |> flank_upstream(4)
```

---

### Chained flanking of each set of anchors

```r
gi

pin_by("first") |> flank_left(2) |

pin_by("second") |> flank_right(2)
```
## S3 method for class 'PinnedGInteractions'
shift_upstream(x, shift)

shift_right(x, shift)

## S3 method for class 'Ranges'
shift_right(x, shift)

## S3 method for class 'PinnedGInteractions'
shift_right(x, shift)

shift_left(x, shift)

## S3 method for class 'Ranges'
shift_left(x, shift)

## S3 method for class 'PinnedGInteractions'
shift_left(x, shift)

### Arguments

- **x**
  
  A PinnedGInteractions object

- **shift**
  
  The amount to move the genomic interval in the Ranges object by. Either a non-negative integer vector of length 1 or an integer vector the same length as x.

### Value

A PinnedGInteractions object

### Examples

```r
# 1. Simple shifting

gi <- read.table(text = "
chr1 11 20 chr1 21 30 + +
chr1 11 20 chr1 51 55 + +
chr1 11 30 chr1 51 55 - -
chr1 11 30 chr2 51 60 - -",
col.names = c("seqnames1", "start1", "end1",
"seqnames2", "start2", "end2", "strand1", "strand2")
) |> as_ginteractions() |> mutate(score = runif(4), type = c('cis', 'cis', 'cis', 'trans'))
```

```r
# 1. Simple shifting

gi
```
gi |> pin_by("first") |> shift_left(15)

gi |> pin_by("second") |> shift_downstream(10)

# 2. Chained shifting of each set of anchors

#|> pin_by("first") |> shift_downstream(20) |> pin_by("second") |> shift_upstream(20)

---

**plyranges-stretch**

*Stretch pinned anchors of a GInteractions object with plyranges*

**Description**

Stretch pinned anchors of a GInteractions object with plyranges

**Usage**

```r
## S3 method for class 'AnchoredPinnedGInteractions'
stretch(x, extend)

## S3 method for class 'PinnedGInteractions'
stretch(x, extend)
```

**Arguments**

- `x` a PinnedGInteractions object
- `extend` The amount to alter the width of a Ranges object by. Either an integer vector of length 1 or an integer vector the same length as `x`.

**Value**

A PinnedGInteractions object

**Examples**

```r
gi <- read.table(text = "
chr1 11 20 chr1 21 30 + +
chr1 11 20 chr1 51 55 + +
chr1 11 30 chr1 51 55 - -
chr1 11 30 chr2 51 60 - -",
col.names = c("seqnames1", "start1", "end1",
"seqnames2", "start2", "end2", "strand1", "strand2")
) |> as_ginteractions() |
```
mutate(score = runif(4), type = c('cis', 'cis', 'cis', 'trans'))

# 1. Simple stretching

## 1. Simple stretching

gi

<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
</tr>
</tbody>
</table>

---

# 2. Chained stretching of each set of anchors

## 2. Chained stretching of each set of anchors

<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
</tr>
</tbody>
</table>

===

---

**Description**

These objects are imported from other packages. Follow the links below to see their documentation.

**dplyr** arrange, count, filter, group_by, group_data, group_indices, group_keys, group_rows, group_size, group_vars, groups, mutate, n_groups, rename, select, slice, summarise, summarize, tally, ungroup

**plyranges** anchor, anchor_3p, anchor_5p, anchor_center, anchor_end, anchor_start, count_overlaps, count_overlaps_directed, filter_by_non_overlaps, filter_by_overlaps, find_overlaps, find_overlaps_directed, join_overlap_left, join_overlap_left_directed, stretch, unanchor

**Value**

Depending on the re-exported function

**Examples**

```r
1 + 1
```
**replace_anchors**

*Replace anchors of a GInteractions*

**Description**

Replace anchors of a GInteractions

**Usage**

```r
replace_anchors(x, id, value)
```

## S4 method for signature 'GInteractions,character,GenomicRanges'

```r
replace_anchors(x, id, value)
```

## S4 method for signature 'GInteractions,numeric,GenomicRanges'

```r
replace_anchors(x, id, value)
```

## S4 method for signature 'PinnedGInteractions,missing,GenomicRanges'

```r
replace_anchors(x, id, value)
```

## S4 method for signature 'AnchoredPinnedGInteractions,missing,GRanges'

```r
replace_anchors(x, id, value)
```

## S4 method for signature 'AnchoredPinnedGInteractions,numeric,GRanges'

```r
replace_anchors(x, id, value)
```

**Arguments**

- **x**
  - a (Pinned)GInteractions object

- **id**
  - Which anchors to replace ("first" or "second"). Ignored if the GInteractions is already pinned to a specific set of anchors.

- **value**
  - A GRanges object vector the same length as x.

**Value**

a (Pinned)GInteractions object.

**Examples**

```r
gi <- read.table(text = "
chr1 11 20 chr1 21 30
chr1 11 20 chr1 51 55
chr1 11 30 chr1 51 55
chr1 11 30 chr2 51 60",
col.names = c(
  "seqnames1", "start1", "end1",
  "seqnames2", "start2", "end2")
```

```r
```
```r
# Internal GInteractions setters

## Description

Internal GInteractions setters

## Usage

- `set_seqnames1(x, value)`
- `set_seqnames2(x, value)`
- `set_start1(x, value)`
```r
set_seqnames1(x, value)
set_start2(x, value)
set_end1(x, value)
set_end2(x, value)
set_width1(x, value)
set_width2(x, value)
set_strand1(x, value)
set_strand2(x, value)

## S4 replacement method for signature 'GInteractions'
first(x) <- value

## S4 replacement method for signature 'GInteractions'
second(x) <- value

## S4 method for signature 'GInteractions,factor'
set_seqnames1(x, value)

## S4 method for signature 'GInteractions,factor'
set_seqnames2(x, value)

## S4 method for signature 'GInteractions,numeric'
set_start1(x, value)

## S4 method for signature 'GInteractions,numeric'
set_start2(x, value)

## S4 method for signature 'GInteractions,numeric'
set_end1(x, value)

## S4 method for signature 'GInteractions,numeric'
set_end2(x, value)

## S4 method for signature 'GInteractions,numeric'
set_width1(x, value)

## S4 method for signature 'GInteractions,numeric'
set_width2(x, value)

## S4 method for signature 'AnchoredPinnedGInteractions,numeric'
set_width1(x, value)

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

set_width2(x, value)

## S4 method for signature 'GInteractions,character'
set_strand1(x, value)

## S4 method for signature 'GInteractions,character'
set_strand2(x, value)

**Arguments**

- **x**  
  a GInteractions object

- **value**  
  a value passed to the corresponding field

**Value**

A modified GInteractions
Index

* internal
  ce10_ARCC, 10
  ce10_RESs, 11
  delegating-ginteractions-methods, 11
  GM12878_HiCCUPS, 33
  reexports, 42
  set_seqnames1, 44
$,GInteractions-method (anchors1), 4
anchor, 42
anchor (reexports), 42
anchor_AnchoredPinnedGInteractions (ginteractions-anchor), 24
anchor_3p, 42
anchor_3p (reexports), 42
anchor_3p_AnchoredPinnedGInteractions (ginteractions-anchor), 24
anchor_3p_PinnedGInteractions (ginteractions-anchor), 24
anchor_5p, 42
anchor_5p (reexports), 42
anchor_5p_AnchoredPinnedGInteractions (ginteractions-anchor), 24
anchor_5p_PinnedGInteractions (ginteractions-anchor), 24
anchor_center, 42
anchor_center (reexports), 42
anchor_center_AnchoredPinnedGInteractions (ginteractions-anchor), 24
anchor_center_PinnedGInteractions (ginteractions-anchor), 24
anchor_end, 42
anchor_end (reexports), 42
anchor_end_AnchoredPinnedGInteractions (ginteractions-anchor), 24
anchor_end_PinnedGInteractions (ginteractions-anchor), 24
anchor_start, 42
anchor_start (reexports), 42
anchor_start_AnchoredPinnedGInteractions (ginteractions-anchor), 24
anchor_start_PinnedGInteractions (ginteractions-anchor), 24
anchors, DelegatingGInteractions-method (delegating-ginteractions-methods), 11
anchors1, 4
anchors1_DelegatingGInteractions-method (delegating-ginteractions-methods), 11
anchors1_GInteractions-method (anchors1), 4
annotate, 6
annotate_GInteractions, GRanges, character-method (annotate), 6
annotate_directed (annotate), 6
annotate_directed_GInteractions, GRanges, character-method (annotate), 6
arrange, 3, 42
arrange (reexports), 42
arrange_GInteractions (dplyr-arrange), 13
as_ginteractions, 8
ce10_ARCC, 10
ce10_RESs, 11
count, 3, 42
count (reexports), 42
count_GInteractions (dplyr-count), 14
count_GroupedGInteractions (dplyr-count), 14
count_overlaps, 42
count_overlaps (reexports), 42
INDEX

count_overlaps.GInteractions
  (ginteractions-count-overlaps), 26

count_overlaps.PinnedGInteractions
  (ginteractions-count-overlaps), 26

count_overlaps_directed, 42

count_overlaps_directed (reexports), 42

count_overlaps_directed.GInteractions
  (ginteractions-count-overlaps), 26

count_overlaps_directed.PinnedGInteractions
  (ginteractions-count-overlaps), 26

count_overlaps, 26, 28, 32

data.frame(), 8

DataFrame(), 8, 15, 23

dplyr-arrange, 13
dplyr-count, 14
dplyr-filter, 15
dplyr-group_by, 16
dplyr-mutate, 18
dplyr-rename, 20

dplyr-select, 21
dplyr-slice, 22

end1 (anchors1), 4

end1,DelegatingGInteractions-method
  (delegating-ginteractions-methods), 11

end1,GInteractions-method (anchors1), 4

dend2 (anchors1), 4

end2,DelegatingGInteractions-method
  (delegating-ginteractions-methods), 11

end2,GInteractions-method (anchors1), 4

filter, 3, 42

filter (reexports), 42

filter.GInteractions (dplyr-filter), 15

filter_by_non_overlaps, 42

filter_by_non_overlaps (reexports), 42

filter_by_non_overlaps.GInteractions
  (ginteractions-filter-overlaps), 27

filter_by_non_overlaps.PinnedGInteractions
  (ginteractions-filter-overlaps), 27

filter_by_overlaps, 42

filter_by_overlaps (reexports), 42

filter_by_overlaps.GInteractions
  (ginteractions-filter-overlaps), 27

filter_by_overlaps.PinnedGInteractions
  (ginteractions-filter-overlaps), 27

find_overlaps, 42

find_overlaps (reexports), 42

find_overlaps.GInteractions
  (ginteractions-find-overlaps), 29

find_overlaps.PinnedGInteractions
  (ginteractions-find-overlaps), 29

find_overlaps_directed, 42

find_overlaps_directed (reexports), 42

find_overlaps_directed.GInteractions
  (ginteractions-find-overlaps), 29

find_overlaps_directed.PinnedGInteractions
  (ginteractions-find-overlaps), 29

findOverlaps, 30

first<-,GInteractions-method
  (set_seqnames1), 44

flank_downstream (plyranges-flank), 37

flank_left (plyranges-flank), 37

flank_right (plyranges-flank), 37

flank_upstream (plyranges-flank), 37

ginteractions-anchor, 24
ginteractions-annotate (annotate), 6
ginteractions-count-overlaps, 26
ginteractions-filter-overlaps, 27
ginteractions-getters (anchors1), 4
ginteractions-join-overlap-left, 31
ginteractions-pin (pin), 35

ginteractions-setters (set_seqnames1), 44

GM12878_HiCCUPS, 33
group-group_data, 34

group_by, 3, 42
group_by (reexports), 42
group_by.DelegatingGInteractions
    (dplyr-group_by), 16

join_overlap_left_directed.PinnedGInteractions
    (ginteractions-jion-overlap-left), 31

mcols,DelegatingGInteractions-method
    (delegating-ginteractions-methods), 11

mutate, 3, 42
mutate (reexports), 42
mutate.GInteractions (dplyr-mutate), 18

n_groups, 42
n_groups (reexports), 42
n_groups.GroupedGInteractions
    (group-group_data), 34

pin, 35
pin,AnchoredPinnedGInteractions,character-method
    (pin), 35
pin,AnchoredPinnedGInteractions,numeric-method
    (pin), 35
pin,GInteractions,character-method
    (pin), 35
pin,GInteractions,numeric-method (pin), 35
pin,GroupedGInteractions,character-method
    (pin), 35
pin,GroupedGInteractions,numeric-method
    (pin), 35
pin,PinnedGInteractions,character-method
    (pin), 35
pin,PinnedGInteractions,missing-method
    (pin), 35
pin,PinnedGInteractions,numeric-method
    (pin), 35
pin_anchors1 (pin), 35
pin_anchors2 (pin), 35
pin_by (pin), 35
pin_first (pin), 35
pin_second (pin), 35
pinned_anchors (pin), 35
pinned_anchors,AnchoredPinnedGInteractions-method
    (pin), 35
pinned_anchors,GroupedGInteractions-method
    (pin), 35
pinned_anchors,PinnedGInteractions-method
    (pin), 35

plyinteractions
    (plyinteractions-package), 3
plyinteractions-package, 3
plyranges-flank, 37
plyranges-shift, 39
plyranges-stretch, 41
ranges1 (anchors1), 4
ranges1, DelegatingGInteractions-method
(delegating-ginteractions-methods), 11
ranges1, GInteractions-method
(anchors1), 4
ranges2 (anchors1), 4
ranges2, DelegatingGInteractions-method
(delegating-ginteractions-methods), 11
ranges2, GInteractions-method
(anchors1), 4
reexports, 42
regions, DelegatingGInteractions-method
(delegating-ginteractions-methods), 11
rename, 42
rename (reexports), 42
rename, GInteractions (dplyr-rename), 20
replace_anchors, 43
replace_anchors, AnchoredPinnedGInteractions, missing, GenomicRanges-method
(replace_anchors), 43
replace_anchors, AnchoredPinnedGInteractions, numeric, GenomicRanges-method
(replace_anchors), 43
replace_anchors, GInteractions, character, GenomicRanges-method
(replace_anchors), 43
replace_anchors, GInteractions, numeric, GenomicRanges-method
(replace_anchors), 43
replace_anchors, PinnedGInteractions, missing, GenomicRanges-method
(replace_anchors), 43
second, GInteractions-method
(set_seqnames1), 44
select, 3, 42
select (reexports), 42
select, GInteractions (dplyr-select), 21
seqinfo, DelegatingGInteractions-method
(delegating-ginteractions-methods), 11
seqnames1 (anchors1), 4
seqnames1, DelegatingGInteractions-method
(delegating-ginteractions-methods), 11
seqnames1, GInteractions-method
(anchors1), 4
seqnames2 (anchors1), 4
seqnames2, DelegatingGInteractions-method
(delegating-ginteractions-methods), 11
seqnames2, GInteractions-method
(anchors1), 4
set_end1 (set_seqnames1), 44
set_end1, GInteractions, numeric-method
(set_seqnames1), 44
set_end2 (set_seqnames1), 44
set_end2, GInteractions, numeric-method
(set_seqnames1), 44
set_seqnames1, 44
set_seqnames1, GInteractions, factor-method
(set_seqnames1), 44
set_seqnames2 (set_seqnames1), 44
set_seqnames2, GInteractions, factor-method
(set_seqnames1), 44
set_start1 (set_seqnames1), 44
set_start1, GInteractions, numeric-method
(set_seqnames1), 44
set_start2 (set_seqnames2), 44
set_start2, GInteractions, numeric-method
(set_seqnames1), 44
set_strand1 (set_seqnames1), 44
set_strand1, GInteractions, character-method
(set_seqnames1), 44
set_strand2 (set_seqnames1), 44
set_strand2, GInteractions, character-method
(set_seqnames1), 44
set_width1, AnchoredPinnedGInteractions, numeric-method
(set_seqnames1), 44
set_width1, GInteractions, numeric-method
(set_seqnames1), 44
set_width2 (set_seqnames1), 44
set_width2, AnchoredPinnedGInteractions, numeric-method
(set_seqnames1), 44
set_width2, GInteractions, numeric-method
(set_seqnames1), 44
shift_downstream (plyranges-shift), 39
shift_left (plyranges-shift), 39
shift_right (plyranges-shift), 39
shift_upstream (plyranges-shift), 39
show, DelegatingGInteractions-method
(delegating-ginteractions-methods), 11
slice, 3, 42
slice (reexports), 42
slice.GInteractions (dplyr-slice), 22
start1 (anchors1), 4
start1, DelegatingGInteractions-method
(delegating-ginteractions-methods), 11
start1, GInteractions-method (anchors1), 4
start2 (anchors1), 4
start2, DelegatingGInteractions-method
(delegating-ginteractions-methods), 11
start2, GInteractions-method (anchors1), 4
strand1 (anchors1), 4
strand1, DelegatingGInteractions-method
(delegating-ginteractions-methods), 11
strand1, GInteractions-method (anchors1), 4
strand2 (anchors1), 4
strand2, DelegatingGInteractions-method
(delegating-ginteractions-methods), 11
strand2, GInteractions-method (anchors1), 4
stretch, 42
stretch (reexports), 42
stretch, AnchoredPinnedGInteractions
(plyranges-stretch), 41
stretch, PinnedGInteractions
(plyranges-stretch), 41
summarise, 42
summarise (reexports), 42
summarise, GroupedGInteractions
(dplyr-summarize), 23
summarize, 3, 42
summarize (reexports), 42
summarize, GroupedGInteractions
(dplyr-summarize), 23
tally, 3, 42
tally (reexports), 42
tally, GroupedGInteractions
(dplyr-count), 14
unanchor, 42
unanchor (reexports), 42
unanchor, AnchoredPinnedGInteractions
(ginteractions-anchor), 24
unanchor, AnchoredPinnedGInteractions
(pin), 35
unpin, PinnedGInteractions-method (pin), 35
width1 (anchors1), 4
width1, DelegatingGInteractions-method
(delegating-ginteractions-methods), 11
width1, GInteractions-method (anchors1), 4
width2 (anchors1), 4
width2, DelegatingGInteractions-method
(delegating-ginteractions-methods), 11
width2, GInteractions-method (anchors1), 4