Package ‘Structstrings’

May 2, 2024

Title  Implementation of the dot bracket annotations with Biostrings
Version  1.20.0
Date  2024-03-19

Description  The Structstrings package implements the widely used dot bracket annotation for storing base pairing information in structured RNA. Structstrings uses the infrastructure provided by the Biostrings package and derives the DotBracketString and related classes from the BString class. From these, base pair tables can be produced for in depth analysis. In addition, the loop indices of the base pairs can be retrieved as well. For better efficiency, information conversion is implemented in C, inspired to a large extend by the ViennaRNA package.

License  Artistic-2.0
Encoding  UTF-8
LazyData  false

biocViews  DataImport, DataRepresentation, Infrastructure, Sequencing, Software, Alignment, SequenceMatching

Depends  R (>= 4.0), S4Vectors (>= 0.27.12), IRanges (>= 2.23.9), Biostrings (>= 2.57.2)

LinkingTo  IRanges, S4Vectors
Imports  methods, BiocGenerics, XVector, stringr, stringi, crayon, grDevices

Suggests  testthat, knitr, rmarkdown, tRNAscanImport, BiocStyle

VignetteBuilder  knitr

RoxygenNote  7.3.1

Collate  'Structstrings.R' 'AllGenerics.R'
'Structstrings-DotBracket-io.R'
'Structstrings-DotBracketDataFrame.R'
'Structstrings-DotBracketString.R'
'Structstrings-DotBracketStringSet.R'
'Structstrings-DotBracketStringSetList.R'
'Structstrings-LoopIndexList.R'
'Structstrings-StructuredXStringSet.R'
Structstrings-package

'Structstrings-alphabet.R' 'Structstrings-conversion.R'
'utils.R' 'zzz.R'

NeedsCompilation yes

BugReports https://github.com/FelixErnst/Structstrings/issues

URL https://github.com/FelixErnst/Structstrings
git_url https://git.bioconductor.org/packages/Structstrings
git_branch RELEASE_3_19
git_last_commit 18cf3f2
git_last_commit_date 2024-04-30
Repository Bioconductor 3.19
Date/Publication 2024-05-01

Author Felix G.M. Ernst [aut, cre] (<https://orcid.org/0000-0001-5064-0928>)
Maintainer Felix G.M. Ernst <felix.gm.ernst@outlook.com>

Contents

Structstrings-package .................................................. 2
convertAnnotation ..................................................... 3
DotBracketDataFrame ................................................... 4
DotBracketString ....................................................... 5
DotBracketStringSet-io .................................................. 6
getBasePairing .......................................................... 8
LoopIndexList .......................................................... 10
Structstrings ........................................................... 11
Structstrings-data ..................................................... 12
Structstrings-internals ............................................... 12
StructuredXStringSet ................................................. 14

Index ................................................................. 16

Structstrings-package

Structstrings: Implementation of the dot bracket annotations with Biostrings

Description

The Structstrings package implements the widely used dot bracket annotation for storing base pairing information in structured RNA. Structstrings uses the infrastructure provided by the Biostrings package and derives the DotBracketString and related classes from the BString class. From these, base pair tables can be produced for in depth analysis. In addition, the loop indices of the base pairs can be retrieved as well. For better efficiency, information conversion is implemented in C, inspired to a large extent by the ViennaRNA package.
**convertAnnotation**

**Author(s)**

Maintainer: Felix G.M. Ernst <felix.gm.ernst@outlook.com> (ORCID)

**See Also**

Useful links:
- [https://github.com/FelixErnst/Structstrings](https://github.com/FelixErnst/Structstrings)
- Report bugs at [https://github.com/FelixErnst/Structstrings/issues](https://github.com/FelixErnst/Structstrings/issues)

---

**convertAnnotation**

*Convert between dot bracket annotations*

**Description**

`convertAnnotation` converts a type of dot bracket annotation into another. This only works if the original bracket type is present and the target bracket type is not.

**Usage**

```r
convertAnnotation(x, from, to)
```

## S4 method for signature 'DotBracketString'

```r
class(convertAnnotation(DotBracketString))
```

## S4 method for signature 'DotBracketStringSet'

```r
class(convertAnnotation(DotBracketStringSet))
```

## S4 method for signature 'DotBracketStringSetList'

```r
class(convertAnnotation(DotBracketStringSetList))
```

**Arguments**

- `x`: a `DotBracketString`, `DotBracketStringSet` or `DotBracketStringSetList`
- `from`: which annotation type should be converted? Must be one of the following values: 1L = '()', 2L = '<>', 3L = '[]', 4L = '{}' and must be present in the input.
- `to`: Into which annotation type should the selected one be converted? Must be one of the following values: 1L = '()', 2L = '<>', 3L = '[]', 4L = '{}'. Must not be present in the input.

**Value**

The modified input object, a `DotBracketString*` object.
Examples

```r
str <- "((.))..[[..]]...(..)........."
dbs <- DotBracketString(str)
convertAnnotation(dbs, 1L, 2L)
```

DotBracketDataFrame  *DataFrame for storing base pairing information*

Description

The DotBracketDataFrame and DotBracketDFrame object is derived from the `DataFrame` and `DFrame` classes. DotBracketDataFrame implements the concept and can be used to implement other backends than the in-memory one as done by DotBracketDFrame.

The DotBracketDataFrameList is implemented analogous, which is also available as CompressedSplitDotBracketDataFrameList. Since the names are quite long, the following short cut functions are available for object creation: DBDF, DBDFL and SDBDFL.

The DotBracketDataFrame can only contain 5 columns, which are named pos, forward, reverse, character and base. The last two columns are optional. The type of the first three has to be integer, whereas the fourth is a character and fifth is a XStringSet column.

Upon creation and modification, the validity of the contained base pairing information is checked. If the information is not correct, an error is thrown.

Usage

```r
DotBracketDataFrame(..., row.names = NULL)
DBDF(...)  
DotBracketDataFrameList(...)  
DBDFL(...)  
SplitDotBracketDataFrameList(..., compress = TRUE, cbindArgs = FALSE)
SDBDFL(..., compress = TRUE, cbindArgs = FALSE)
```

Arguments

- `...` for DotBracketDataFrame the input vectors and for DotBracketDataFrameList the DataFrame or the DotBracketDataFrame objects.
- `row.names` See `DataFrame`
- `compress` If compress = TRUE, returns a CompressedSplitDotBracketDataFrameList else returns a SimpleSplitDotBracketDataFrameList.
- `cbindArgs` If cbindArgs = FALSE, the ... arguments are coerced to DotBracketDataFrame objects and concatenated to form the result. If cbindArgs = TRUE, the arguments are combined as columns. The arguments must then be the same length, with each element of an argument mapping to an element in the result.
DotBracketString

Value

a DotBracketDataFrame* object.

Examples

# Manual creation
df <- DataFrame(pos = c(1,2,3,4,5,6),
               forward = c(6,5,0,0,2,1),
               reverse = c(1,2,0,0,5,6))

# Either works
dbdf <- as(df,"DotBracketDataFrame")
dbdf <- DotBracketDataFrame(df)

# With multiple input DataFrames a SplitDotBracketDataFrameList is returned
bdfl <- DotBracketDataFrame(df,df,df,df)

# Creation from a DotBracketString object is probably more common
data("dbs", package = "Structstrings")
dbdfl <- getBasePairing(dbs)

# Elements are returned as DotBracketDataFrames
dbdfl[[1]]

DotBracketString  The DotBracketString, DotBracketStringSet and DotBracketStringSetList classes

Description

The DotBracketString extends the BString class. The DotBracketStringSet and DotBracketStringSetList classes are implemented accordingly.
The alphabet consists of the letters (,), ...,<,>, [, ], { and }, which describes base pairing between positions. The . letter describes an unpaired position. The number of opening and closing letters need to be equal within a DotBracketString to be a valid dot bracket annotation. This is checked upon creation and modification of the object.
The objects can also be created using the shorter function names DB, DBS and DBSL.
Currently, there is no distinction in base pairing strength between the different bracket types.

Usage

DotBracketString(x = "", start = 1, nchar = NA)

DB(x = character(), start = 1, nchar = NA)

DotBracketStringSet(x = character())

DBS(x = character())

DotBracketStringSetList(..., use.names = TRUE)
DBSL(..., use.names = TRUE)

## S4 method for signature 'DotBracketString'
alphabet(x)

## S4 method for signature 'DotBracketString'
encoding(x)

Arguments

x DotBracketString, DotBracketStringSet: the input, which is tried to be convert into a DotBracketString*.

start DotBracketString: starting position for creating the object from the character input.

nchar DotBracketString: number of letters are read from the input character

... DotBracketStringSetList: the input, which converted into a list. Each element is tried to be converted into a DotBracketStringSet.

use.names DotBracketStringSetList: Should names of the input be preserved.

Value

a DotBracketString* object.

Examples

str <- "((..))..[[..]]...{..}..<<..>>"
db <- DotBracketString(str)
dbs <- DotBracketStringSet(c("structure1" = str, "structure2" = str))
dbsl <- DotBracketStringSetList(list(first = dbs, second = dbs))

---

DotBracketStringSet-io

Reading and writing DotBracketStringSet objects

Description

readDotBracketStringSet and writeDotBracketStringSet are functions to read and write dot bracket strings from/to file. Since the <> is in conflict with the fasta format, saving to fastq file is sometimes the only option. Saving a string with a <> bracket type to a fasta file will throw an error.

The functions use the underlying Biostrings infrastructure and share most of its parameters. For a more detailed look have a look here.
Usage

readDotBracketStringSet(
  filepath,
  format = "fasta",
  nrec = -1L,
  skip = 0L,
  seek.first.rec = FALSE,
  use.names = TRUE,
  with.qualities = FALSE
)

writeDotBracketStringSet(
  x,
  filepath,
  append = FALSE,
  compress = FALSE,
  format = "fasta",
  ...
)

saveDotBracketStringSet(
  x,
  objname,
  dirpath = ".",
  save.dups = FALSE,
  verbose = TRUE
)

Arguments

- **filepath**: The file name, when writing, or file name(s) when reading.
- **format**: "fasta" or "fastq"
- **nrec**: Single integer. The maximum of number of records to read in. Negative values are ignored.
- **skip**: Single non-negative integer. The number of records of the data file(s) to skip before beginning to read in records.
- **seek.first.rec**, **with.qualities**, **compress**, ..., **use.names**, **objname**, **dirpath**, **save.dups**, **verbose**
  Have a look [here](#).
- **x**: A DotBracketStringSet object
- **append**: TRUE or FALSE. If TRUE output will be appended to file. Otherwise, it will overwrite the contents of file.

Value

readDotBracketStringSet returns a DotBracketStringSet object, writeDotBracketStringSet returns NULL invisibly.
getBasePairing

**Examples**

```r
data("dbs", package = "Structstrings")
file <- tempfile()
# works both since a DotBracketStringSet is a BStringSet
writeXStringSet(dbs, file)
writeDotBracketStringSet(dbs, file)
# to return immediately a DotBracketStringSet us readDotBracketStringSet()
dbs2 <- readDotBracketStringSet(file)
```

---

**Description**

getBasePairing converts a dot bracket annotation from a DotBracketString into a base pair table as DotBracketDataFrame. Base pairing is indicated by corresponding numbers in the forward and reverse columns.

getDotBracket converts the dot bracket annotation from a DotBracketDataFrame into a DotBracketString. If the character columns is populated, the information from this column will be used. If this is not desired set force = TRUE. However, beware that this will result in a dot bracket annotation, which does not necessarily matches the original dot bracket string it may have been created from. It is rather the dot bracket string with the lowest number of different loops and it will use the different dot bracket annotations one after another. Example: "((<<>)))" will be returned as (((((()))))). (((<<)))>» will be returned as (((<<)))>», ((([[[[))))]]) will be returned as (((<<)))>».

getLoopIndices converts the dot bracket annotation from a DotBracketString or DotBracketDataFrame into a LoopIndexList.

**Usage**

```r
getBasePairing(x, compress = TRUE, return.sequence = FALSE)
getDotBracket(x, force = FALSE)
getLoopIndices(x, bracket.type, warn.type.drops = TRUE)
```

## S4 method for signature 'DotBracketString'

```r
getBasePairing(x)
```

## S4 method for signature 'DotBracketStringSet'

```r
getBasePairing(x, compress = TRUE)
```

## S4 method for signature 'DotBracketDataFrame'

```r
getDotBracket(x, force = FALSE)
```

## S4 method for signature 'DotBracketDataFrameList'

```r
getDotBracket(x, force = FALSE)
```
getBasePairing

## S4 method for signature 'SimpleSplitDotBracketDataFrameList'
getDotBracket(x, force = FALSE)

## S4 method for signature 'CompressedSplitDotBracketDataFrameList'
getDotBracket(x, force = FALSE)

## S4 method for signature 'DotBracketString'
getLoopIndices(x, bracket.type, warn.type.drops = TRUE)

## S4 method for signature 'DotBracketStringSet'
getLoopIndices(x, bracket.type, warn.type.drops = TRUE)

## S4 method for signature 'DotBracketDataFrame'
getLoopIndices(x, bracket.type, warn.type.drops = TRUE)

## S4 method for signature 'DotBracketDataFrameList'
getLoopIndices(x, bracket.type, warn.type.drops = TRUE)

## S4 method for signature 'SimpleSplitDotBracketDataFrameList'
getLoopIndices(x, bracket.type, warn.type.drops = TRUE)

## S4 method for signature 'CompressedSplitDotBracketDataFrameList'
getLoopIndices(x, bracket.type, warn.type.drops = TRUE)

Arguments

- **x**: a `DotBracketString` or `DotBracketStringSet` object
- **compress**: `getBasePairing`: whether to return a `CompressedSplitDotBracketDataFrameList` or a `SimpleSplitDotBracketDataFrameList`
- **return.sequence**: if the input is a `StructuredXStringSet`: TRUE (default) or FALSE: Whether the sequence should be returned in the base column.
- **force**: `getDotBracket`: Should the dot bracket string be generated from the base pairing, if the character column is present?
- **bracket.type**: `getLoopIndices`: Which dot bracket annotation type should be converted into loop indices? Only usable, if more than one is present. (1L = '()', 2L = '<>', 3L = '[]', 4L = '{}')
- **warn.type.drops**: `getLoopIndices`: TRUE (default) or FALSE: Warn if more than one dot bracket annotation type is present in the input?

Value

- **getBasePairing**: The result is a `DotBracketDataFrame` with following columns: pos, forward, reverse, character (and optionally the base column). If a position is unpaired, forward and reverse will be 0, otherwise it will match the base paired positions.
- **getLoopIndices**: returns a `LoopIndexList`. 
Examples

data("dbs", package = "Structstrings")
# conversion
dbdf <- getBasePairing(dbs)
# ... and the round trip
dbs <- getDotBracket(dbdf)

# loop indices per bracket type
loopids <- getLoopIndices(dbs)
# choose the bracket type manually, if necessary
loopids <- getLoopIndices(dbs, bracket.type = 1L)
# do not show warning if multiple bracket types are present
loopids <- getLoopIndices(dbs, bracket.type = 1L, warn.type.drops = FALSE)

LoopIndexList
LoopIndexList: base pairing information as a list of integer values

Description

With loop indeces base pairing information can be represented by giving each base pair a number
and increasing/decreasing it with each opened/closed base pair. This information can be used for
further analysis of the represented structure.

Usage

LoopIndexList(...)

Arguments

... the integer input vectors.

Value

a LoopIndexList object.

Examples

# if the object is create manually make sure it is a valid structure
# information. Otherwise an error is thrown.
lil <- LoopIndexList(list(c(1L, 2L, 3L, 3L, 2L, 1L, 0L, 5L, 6L, 6L, 5L),
c(1L, 2L, 2L, 2L, 2L, 1L, 0L, 5L, 6L, 6L, 5L)))
Structstrings

Structstrings: implementation of the dot bracket annotations with Biostrings

Description

The Structstrings package implements the widely used dot bracket annotation for storing base pairing information in structured RNA. For example, it is used in the ViennaRNA package (Lorenz et al. 2011), the tRNAscan-SE software (Lowe et al. 1997), and the tRNAdb (Jühling et al. 2009).

Structstrings uses the infrastructure provided by the Biostrings package and derives the class DotBracketString and such from the equivalent BString class. From these base pair tables, one can produce a base pair table for in-depth analysis. For this purpose, the DotBracketDataFrame class is derived from the DataFrame class. In addition, the loop IDs of the base pairs can be retrieved as a LoopIndexList, a derivative of the IntegerList. Generally, it checks automatically for the validity of the dot bracket annotation.

The conversion of the DotBracketString to the base pair table and the loop indices is implemented in C for efficiency. The C implementation is large enough to be inspired by the ViennaRNA package.

This package was developed as a requirement for the tRNA package. However, other projects might benefit as well, so it was split off and improved upon.

Manual

Please refer to the Structstrings vignette for an example of how to work and use the package: Structstrings.

Author(s)

Felix G M Ernst [aut,cre]

References


Structstrings-data  Structstrings example data

Description
Example data for using the Structstrings package

Usage

data(dbs)

data(nseq)

Format

object of class DotBracketStringSet and DNAStringSet
An object of class DNAStringSet of length 299.

Source

sequence and dot bracket annotation of tRNAscan-SE output for *S. cerevisiae* imported using tRNAscanImport. The example file is part of the tRNAscanImport package.

Structstrings-internals  Structstrings internals

Description
Analog to Biostrings there are a few objects, which should only be used internally, but may be of use to other package developers. Otherwise take care.

Usage

DOTBRACKET_CHAR_VALUES
DOTBRACKET_ALPHABET
STRUCTURE_NEUTRAL_CHR
STRUCTURE_OPEN_CHR
STRUCTURE_CLOSE_CHR

## S4 replacement method for signature 'DotBracketDataFrame'
x[i, j, ...] <- value

## S4 replacement method for signature 'CompressedSplitDotBracketDataFrameList'
colnames(x) <- value

## S4 method for signature 'DotBracketString'
seqtype(x)

## S4 method for signature 'DotBracketString'
subseq(x, start = NA, end = NA, width = NA)

## S4 replacement method for signature 'DotBracketString'
subseq(x, start = NA, end = NA, width = NA) <- value

## S4 replacement method for signature 'DotBracketStringSet'
subseq(x, start = NA, end = NA, width = NA) <- value

Arguments

seqtype, x, start, end, width, value, i, j, ...

used internally

Format

a integer vector of length 9 containing the integer values of the dotbracket alphabet
a character vector of length 9 containing the single characters of the dotbracket alphabet
a character vector of length 1 containing the character for unpaired positions
a character vector of length 4 containing the opening character of the dotbracket alphabet
a character vector of length 4 containing the closing character of the dotbracket alphabet

Examples

DOTBRACKET_CHAR_VALUES
DOTBRACKET_ALPHABET
STRUCTURE_NEUTRAL_CHR
STRUCTURE_OPEN_CHR
STRUCTURE_CLOSE_CHR

# the replace method for a DotBracketDataFrame had to be reimplemented
# because of the requirement of columns for a DotBracketDataFrameList and
# DotBracketDataFrame
data("dbs", package = "Structstrings")
dbdfl <- getBasePairing(dbs)
# Elements are returned as DotBracketDataFrames
dbdf <- dbdfl[[1]]
dbdfl[[1]] <- dbdf
dbdf[[1]] <- dbdfl[[1]]
StructuredXStringSet

StructuredRNAStringSet for storing DotBracketAnnotation alongside nucleotide sequences

Description

The StructuredXStringSet class can be used to store structure information alongside RNA sequences. The class behaves like the QualityScaledXStringSet classes.

Please note, that this does not check for validity regarding base pairing capabilities.

Usage

StructuredRNAStringSet(x, structure)

dotbracket(x)

dotbracket(x) <- value

## S4 method for signature 'StructuredXStringSet'
dotbracket(x)

## S4 replacement method for signature 'StructuredXStringSet'
dotbracket(x) <- value

readStructuredRNAStringSet(
  filepath,
  nrec = -1L,
  skip = 0L,
  seek.first.rec = FALSE,
  use.names = TRUE
)

writeStructuredXStringSet(x, filepath, append = FALSE, compress = FALSE, ...)

## S4 method for signature 'StructuredXStringSet'
getBasePairing(x, compress = TRUE, return.sequence = FALSE)

## S4 method for signature 'StructuredXStringSet'
getLoopIndices(x, bracket.type, warn.type.drops = TRUE)

Arguments

x

For the Structured*StringSet constructors: Either a character vector, or an RNAString, RNAStringSet object. For writeStructuredXStringSet: A StructuredRNAStringSet derivative.

structure, value

A DotBracketStringSet
use.names, type, filepath, nrec, skip, seek.first.rec, append, ...
See DotBracketStringSet-io
compress
return.sequence
TRUE(default) or FALSE: Whether the sequence should be returned in the base
column.
bracket.type getLoopIndices: Which dot bracket annotation type should be converted into
loop indices? Only usable, if more than one is present. (1L = '()', 2L = '<>',
3L = '[ ]', 4L = '{}')
warn.type.drops
See getLoopIndices

Details
the dotbracket function allows access to the included DotBracketStringSet.

Value
a StructuredRNAStringSet object.

Examples
str <- DotBracketStringSet("((())")
seq <- RNAStringSet("AGCU")
sdbs <- StructuredRNAStringSet(seq, str)
Index

* datasets
  Structstrings-data, 12
  Structstrings-internals, 12

* internal
  Structstrings-package, 2
  <-, DotBracketDataFrame-method
    (Structstrings-internals), 12
  alphabet, DotBracketString-method
    (DotBracketString), 5
  BString, 5, 11
  colnames<-, CompressedSplitDotBracketDataFrameList-method
    (Structstrings-internals), 12
  CompressedSplitDotBracketDataFrameList-class
    (DotBracketDataFrame), 4
  CompressedSplitDotBracketDFrameList-class
    (DotBracketDataFrame), 4
  convertAnnotation, 3
  convertAnnotation, DotBracketString-method
    (convertAnnotation), 3
  convertAnnotation, DotBracketStringSet-method
    (convertAnnotation), 3
  convertAnnotation, DotBracketStringSetList-method
    (convertAnnotation), 3
  DataFrame, 4, 11
  DB (DotBracketString), 5
  DBDF (DotBracketDataFrame), 4
  DBDFL (DotBracketDataFrame), 4
  DBS (DotBracketString), 5
  dbs (Structstrings-data), 12
  DBSLS (DotBracketString), 5
  DFrame, 4
  DNAStringSet, 12
  dotbracket (StructuredXStringSet), 14
  dotbracket, StructuredXStringSet-method
    (StructuredXStringSet), 14
  dotbracket<-, StructuredXStringSet-method
    (StructuredXStringSet), 14
  DOTBRACKET_ALPHABET
    (Structstrings-internals), 12
  DOTBRACKET_CHAR_VALUES
    (Structstrings-internals), 12
  DotBracketDataFrame, 4, 8, 9, 11
  DotBracketDataFrame-class
    (DotBracketDataFrame), 4
  DotBracketDataFrameList
    (DotBracketDataFrame), 4
  DotBracketDataFrameList-class
    (DotBracketDataFrame), 4
  DotBracketDFrame-class
    (DotBracketDataFrame), 4
  DotBracketDFrameList-class
    (DotBracketDataFrame), 4
  DotBracketString, 5, 8, 9, 11
  DotBracketString-class
    (DotBracketString), 5
  DotBracketStringSet, 9, 12, 14, 15
  DotBracketStringSet (DotBracketString), 5
  DotBracketStringSet-class
    (DotBracketString), 5
  DotBracketStringSet-listio, 6
  DotBracketStringSet-class
    (DotBracketString), 5
  encoding, DotBracketString-method
    (DotBracketString), 5
  getBasePairing, 8, 15
  getBasePairing, DotBracketString-method
    (getBasePairing), 8
  getBasePairing, DotBracketStringSet-method
    (getBasePairing), 8
getBasePairing, StructuredXStringSet-method (StructuredXStringSet), 14
getDotBracket (getBasePairing), 8
getDotBracket, CompressedSplitDotBracketDataFrameList-method (getBasePairing), 8
getDotBracket, DotBracketDataFrame-method (getBasePairing), 8
getDotBracket, DotBracketDataFrameList-method (getBasePairing), 8
getDotBracket, SimpleSplitDotBracketDataFrameList-method (getBasePairing), 8
getLoopIndices (getBasePairing), 8
getLoopIndices, CompressedSplitDotBracketDataFrameList-method (getBasePairing), 8
getLoopIndices, DotBracketDataFrame-method (getBasePairing), 8
getLoopIndices, DotBracketDataFrameList-method (getBasePairing), 8
getLoopIndices, SimpleSplitDotBracketDataFrameList-method (getBasePairing), 8
getLoopIndices, StructuredXStringSet-method (StructuredXStringSet), 14

SimpleSplitDotBracketDataFrameList-class (DotBracketDataFrame), 4
SimpleSplitDotBracketDataFrameList-class (DotBracketDataFrame), 4
SplitDotBracketDataFrameList (DotBracketDataFrame), 4
StructuredRNAStringSet (StructuredXStringSet), 14
StructuredRNAStringSet-class (StructuredXStringSet), 14
StructuredRNAStringSet-class (StructuredXStringSet), 14

subseq, DotBracketString-method (Structstrings-internals), 12
subseq<-, DotBracketString-method (Structstrings-internals), 12
subseq<-, DotBracketStringSet-method (Structstrings-internals), 12

writeDotBracketStringSet (DotBracketStringSet-io), 6
writeStructuredXStringSet (StructuredXStringSet), 14

tRNAscanImport, 12

writeDotBracketStringSet (DotBracketStringSet-io), 6
writeStructuredXStringSet (StructuredXStringSet), 14

here, 6, 7

IntegerList, 11
LoopIndexList, 8, 9, 10, 11
LoopIndexList-class (LoopIndexList), 10

nseq (Structstrings-data), 12

QualityScaledXStringSet, 14

readDotBracketStringSet (DotBracketStringSet-io), 6
readStructuredRNAStringSet (StructuredXStringSet), 14

saveDotBracketStringSet (DotBracketStringSet-io), 6

SDBDFL (DotBracketDataFrame), 4

seqtype, DotBracketString-method (Structstrings-internals), 12