Package ‘scanMiR’

April 4, 2024

Type Package
Title scanMiR
Version 1.8.2
Depends R (>= 4.0)
Date 2024-02-03
Imports Biostrings, GenomicRanges, IRanges, data.table, BiocParallel, methods, GenomeInfoDb, S4Vectors, ggplot2, stats, stringi, utils, graphics, grid, seqLogo, cowplot
Suggests knitr, rmarkdown, BiocStyle, testthat (>= 3.0.0)
Description A set of tools for working with miRNA affinity models (KdModels), efficiently scanning for miRNA binding sites, and predicting target repression. It supports scanning using miRNA seeds, full miRNA sequences (enabling 3’ alignment) and KdModels, and includes the prediction of slicing and TDMD sites. Finally, it includes utility and plotting functions (e.g. for the visual representation of miRNA-target alignment).
License GPL-3
VignetteBuilder knitr
RoxygenNote 7.2.3
biocViews miRNA, SequenceMatching, Alignment
Config/testthat/edition 3
git_url https://git.bioconductor.org/packages/scanMiR
git_branch RELEASE_3_18
git_last_commit 6400e75
git_last_commit_date 2024-02-11
Repository Bioconductor 3.18
Date/Publication 2024-04-03
Author Pierre-Luc Germain [cre, aut] (<https://orcid.org/0000-0003-3418-4218>), Michael Soutschek [aut], Fridolin Gross [aut]
Maintainer Pierre-Luc Germain <pierre-luc.germain@hest.ethz.ch>
aggregateMatches

Description

Aggregates miRNA binding sites with log_kd values to predict transcript repression. See the vignette for more detail.

Usage

```r
aggregateMatches(
  m,
  a = 0.007726,
  b = 0.5735,
  c = 0.181,
  p3 = 0.051,
  coef_utr = 0,
  coef_orf = 0,
  p3.range = c(3L, 8L),
  keepSiteInfo = TRUE,
  toItem = FALSE,
  BP = NULL
)
```
assignKdType

Arguments

m  A GRanges or data.frame of matches as returned by ‘findSeedMatches’.
a  The relative concentration of unbound AGO-miRNA complexes.
b  Factor specifying the additional repression by a single bound AGO.
c  Penalty for sites that are found within the ORF region.
p3  Factor specifying additional repression due to 3p alignment.
coef_utr  Factor specifying additional repression due to UTR length.
coef_orf  Factor specifying additional repression due to ORF length.
p3.range  Range used for 3p alignment.
keepSiteInfo  Logical; whether to return information about site types (default = TRUE). Ignored if ‘m’ does not contain ‘log_kd’ values
toInt  Logical; whether to convert repression scores to integers (default = FALSE).
BP  Pass ‘BiocParallel::MulticoreParam(ncores, progressbar=TRUE)’ to enable multithreading. Note that in addition, ‘aggregateMatches’ uses the data.table package, which is often set to use multi-threading by default (which would be multiplied by threads determined by ‘BP’). See setDTthreads for more information.

Value

a data.frame containing aggregated repression values and/or information about the numbers and types of matches

Examples

# we create mock RNA sequences and seeds:
segs <- getRandomSeq(n=10)

# load sample KdModel
data(SampleKdModel)

# find matches
matches <- findSeedMatches(seqs, SampleKdModel)

# aggregate matches
aggregateMatches(matches)

assignKdType assignKdType

Description

Assigns a log_kd and match type to a set of matched sequences.

Usage

assignKdType(x, mod, mer8 = NULL)
conservation

Arguments

x  A vector of matched sequences, each of 12 nucleotides
mod  An object of class 'KdModel'
mer8  The optional set of 8mers included in the model (for internal use; can be reconstructed from the model).

Value

A data.frame with one row for each element of 'x', and the columns 'type' and 'log_kd'. To save space, the reported log_kd is multiplied by 1000, rounded and saved as an integer.

Examples

data(SampleKdModel)
assignKdType(c("CTAGCATTAAGT","ACGTACGTACGT"), SampleKdModel)

conservation

Description

conservation

Usage

conservation(x)

Arguments

x  A KdModelList, or a KdModel

Value

A vector of the conservation status for each miRNA

Examples

data(SampleKdModel)
conservation(SampleKdModel)
dummyKdData

Create dummy log\_kd per 12-mer data

**Description**

Create dummy log\_kd per 12-mer data

**Usage**

dummyKdData(mod = NULL)

**Arguments**

mod Optional model from which to create the dummy data

**Value**

A data.frame with 12-mers and log\_kds

**Examples**

kd <- dummyKdData()

findSeedMatches

Predicting and characterizing miRNA binding sites

**Description**

‘findSeedMatches’ takes a set of sequences and a set of miRNAs (given either as target seeds, mature miRNA sequences, or a KdModelList).

**Usage**

findSeedMatches(
    seqs,
    seeds,
    shadow = 0L,
    onlyCanonical = FALSE,
    maxLogKd = c(-1, -1.5),
    keepMatchSeq = FALSE,
    minDist = 7L,
    p3.extra = FALSE,
    p3.params = list(maxMirLoop = 7L, maxTargetLoop = 9L, maxLoopDiff = 4L, mismatch = TRUE, GUwob = TRUE),
    agg.params = .defaultAggParams(),
    ret = c("GRanges", "data.frame", "aggregated"),
)
Arguments

seqs A character vector or ‘DNAStringSet’ of DNA sequences in which to look.

seeds A character vector of 7-nt seeds to look for. If RNA, will be reversed and complemented before matching. If DNA, they are assumed to be the target sequence to look for. Alternatively, a list of objects of class ‘KdModel’ or an object of class ‘KdModelList’ can be given.

shadow Integer giving the shadow, i.e. the number of nucleotides hidden at the beginning of the sequence (default 0).

onlyCanonical Logical; whether to restrict the search only to canonical binding sites.

maxLogKd Maximum log_kd value to keep. This has a major impact on the number of sites returned, and hence on the memory requirements. Set to Inf to disable (_not_ recommended when running large scans!).

keepMatchSeq Logical; whether to keep the sequence (including flanking dinucleotides) for each seed match (default FALSE).

minDist Integer specifying the minimum distance between matches of the same miRNA (default 7). Closer matches will be reduced to the highest-affinity. To disable the removal of overlapping features, use ‘minDist=-Inf’.

p3.extra Logical; whether to keep extra information about 3’ alignment. Disable (default) this when running large scans, otherwise you might hit your system’s memory limits.

p3.params Named list of parameters for 3’ alignment with slots ‘maxMirLoop’ (integer, default = 7), ‘maxTargetLoop’ (integer, default = 9), ‘maxLoopDiff’ (integer, default = 4), ‘mismatch’ (logical, default = TRUE) and ‘GUwob’ (logical, default = TRUE).

agg.params A named list with slots ‘a’, ‘b’, ‘c’, ‘p3’, ‘coef_utr’, ‘coef_orf’ and ‘keepSiteInfo’ indicating the parameters for the aggregation. Ignored if ‘ret=”aggregated”’. For further details see documentation of ‘aggregateMatches’.

ret The type of data to return, either "GRanges" (default), "data.frame", or "aggregated" (aggregates affinities/sites for each seed-transcript pair).

BP Pass ‘BiocParallel::MulticoreParam(ncores, progressbar=TRUE)’ to enable multithreading.

verbose Logical; whether to print additional progress messages (default on if not multithreading)

n_seeds Integer; the number of seeds that are processed in parallel to avoid memory issues.
**get3pAlignment**

**Value**

A GRanges of all matches. If `seeds` is a `KdModel` or `KdModelList`, the `log_kd` column will report the ln(Kd) multiplied by 1000, rounded and saved as an integer. If `ret!="GRanges`, returns a data.frame.

**Examples**

```r
# we create mock RNA sequences and seeds:
seqs <- getRandomSeq(n=10)
seeds <- c("AAACCAC", "AAACCUU")
findSeedMatches(seqs, seeds)
```

---

**get3pAlignment**  
*Finds 3’ complementary binding of a miRNA*

**Description**

Performs a local alignment of the miRNA 3’ sequence (determined by `mir3p.start`) on given the given sequences.

**Usage**

```r
get3pAlignment(
  seqs,
  mirseq,
  mir3p.start = 9L,
  allow.mismatch = TRUE,
  maxMirLoop = 7L,
  maxTargetLoop = 9L,
  maxLoopDiff = 4L,
  TGsub = TRUE,
  siteType = NULL
)
```
Arguments

seqs         A set of sequences in which to look for 3' matches (i.e. upstream of the seed match)
mirseq       The sequence of the mature miRNA
mir3p.start  The position in 'mirseq' in which to start looking
allow.mismatch Logical; whether to allow mismatches
maxMirLoop   Maximum miRNA loop size
maxTargetLoop Maximum target loop size
maxLoopDiff  Maximum size difference between miRNA and target loops
TGSsub       Logical; whether to allow T/G substitutions.
siteType     The optional type of seed-complementarity, as returned by `getMatchTypes`. This is needed to identify slicing/TDMD sites. If given, should be a vector of the same length as 'seqs'.

Value

A data.frame with one row for each element of 'seqs', indicating the size of the miRNA bulge, the size of the target mRNA bulge, the number of mismatches at the 3' end, and the partial 3' alignment score (i.e. roughly the number of consecutive matching nucleotides)

Examples

```
get3pAlignment(seqs="NNAGTGCTCCATNN", mirseq="TGGAGTGCTGACAATGCTTGG")
```

**Description**

Returns the minimum and maximum 8-mer log-kd values

**Usage**

```
get8merRange(mod)
```

**Arguments**

mod           A 'KdModel'

**Value**

A numeric vector of length two

**Examples**

```
data("SampleKdModel")
get8merRange(SampleKdModel)
```
**getKdModel**

**Description**

getKdModel

**Usage**

getKdModel(kd, mirseq = NULL, name = NULL, conservation = NA_integer_, ...)

**Arguments**

kd
   A data.frame containing the log_kd per 12-mer sequence, or the path to a text/csv file containing such a table. Should contain the columns 'log_kd', '12mer' (or 'X12mer'), and eventually 'mirseq' (if the 'mirseq' argument is NULL) and 'mir' (if the 'name' argument is NULL).

mirseq
   The miRNA (cDNA) sequence.

name
   The name of the miRNA.

conservation
   The conservation level of the miRNA. See 'scanMiR:::.conservation_levels()' for possible values.

...
   Any additional information to be saved with the model.

**Value**

An object of class 'KdModel'.

**Examples**

```r
dk <- dummyKdData()
mod <- getKdModel(kd=kd, mirseq="TTAATGCTAATCGTGATAGGGGTT", name="my-miRNA")
```

**getKmers**

**Description**

Returns all combinations of 'n' elements of 'from'

**Usage**

getKmers(n = 4, from = c("A", "C", "G", "T"))
getMatchTypes

Arguments

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>n</td>
<td>Number of elements from Letters sampled</td>
</tr>
</tbody>
</table>

Value

A character vector

Examples

```r
getKmers(3)
```

---

getMatchTypes  getMatchTypes

Description

Given a seed and a set of sequences matching it, returns the type of match.

Usage

```r
getMatchTypes(x, seed, checkWobble = TRUE)
```

Arguments

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>x</td>
<td>A character vector of short sequences.</td>
</tr>
<tr>
<td>seed</td>
<td>A 7 or 8 nucleotides string indicating the seed (5' to 3' sequence of the target RNA). If of length 7, an &quot;A&quot; will be appended.</td>
</tr>
<tr>
<td>checkWobble</td>
<td>Whether to flag wobbled sites</td>
</tr>
</tbody>
</table>

Value

A factor of match types.

Examples

```r
x <- c("AACACTCCAG", "GACACTCCGC", "GTACTCCAT", "ACGTACGTAC")
getMatchTypes(x, seed="ACACTCCA")
```
**Description**

Produces a random sequence of the given letters

**Usage**

```r
generateRandomSeq(length = 3000, alphabet = c("A", "C", "G", "T"), n = 1)
```

**Arguments**

- `length`: Length of the sequence
- `alphabet`: Letters from which to sample
- `n`: The number of sequences to generate

**Value**

A character vector of length 1

**Examples**

```r
generateRandomSeq(100)
```

---

**Description**

Generates all possible 8mers with 4 consecutive and positioned matches to a given seed.

**Usage**

```r
generateSeed8mers(seed, addNs = FALSE)
```

**Arguments**

- `seed`: The miRNA seed (target DNA sequence), a character vector of length 8 (if of length 7, a "A" will be added on the right)
- `addNs`: Logical; whether to include 8mers with one flanking N

**Value**

A vector of 1024 8mers.
### Examples

```r
head(getSeed8mers("ACACTCCA"))
```

---

**KdModel**  
*miRNA affinity models*

---

### Description

Methods for the `KdModel` class

### Usage

```r
## S4 method for signature 'KdModel'
show(object)
```

```r
## S4 method for signature 'KdModel'
summary(object)
```

```r
## S4 method for signature 'KdModel'
c(x, ...)
```

### Arguments

- `object, x, ...` An object of class `KdModel`

### Value

Depends on the method.

### See Also

- `KdModel`, `KdModelList`

### Examples

```r
data(SampleKdModel)
SampleKdModel
summary(SampleKdModel)
```
KdModelList-class

Description

KdModelList

Usage

KdModelList(..., description = NULL, makeUnique = FALSE)

Arguments

... Any number of KdModel objects or lists thereof.
  description A description for the collection.
  makeUnique Logical; whether to rename models if names are duplicated.

Value

A KdModelList

Examples

data(SampleKdModel)
mods <- KdModelList(SampleKdModel, SampleKdModel, makeUnique = TRUE)
mods

KdModelList-methods

Methods for the KdModelList classes

Description

Methods for the KdModelList classes

Usage

## S4 method for signature 'KdModelList'
summary(object)

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

Arguments

object, x An object of class KdModelList
i the index of item(s) to select
j, drop, ... ignored
plotKdModel

Value

Depends on the method.

See Also

KdModel, KdModelList

Examples

# create a KdModelList:
data(SampleKdModel)
kml <- KdModelList(SampleKdModel, SampleKdModel, makeUnique=TRUE)
summary(kml)
kml[[1]] # returns a KdModelList
kml[[2]] # returns a KdModel
conservation(kml)

Description

Plots the summary of an affinity model.

Usage

plotKdModel(mod, what = c("both", "seeds", "logo"), n = 10)

Arguments

mod

A ‘KdModel’

what

Either 'seeds', 'logo', or 'both' (default).

n

The number of top 7-mers to plot (when 'what='seeds'')

Details

'what='seeds'‘ plots the -$\log(K_d)$ values of the top ‘n’ 7-mers (including both canonical and non-canonical sites), with or without the final "A" vis-a-vis the first miRNA nucleotide. ‘what='logo’ plots a ‘seqLogo’ (requires the [seqLogo](https://bioconductor.org/packages/release/bioc/html/seqLogo.html) package) showing the nucleotide-wise information content and preferences for all 12-mers (centered around the seed, oriented in the direction of the target mRNA). ‘what="both"’ plots both. Note that if the package ‘ggseqlogo’ is installed, this will be used instead to plot the logo, resulting in more detailed plot annotation.

Value

If ‘what="logo"’, returns nothing and plots a position weight matrix. Otherwise returns a ggplot.
removeOverlappingRanges

Examples

data(SampleKdModel)
plotKdModel(SampleKdModel, what="seeds")

removeOverlappingRanges

removeOverlappingRanges

Description

Removes elements from a GRanges that overlap (or are within a given distance of) other elements higher up in the list (i.e. assumes that the ranges are sorted in order of priority). The function handles overlaps between more than two ranges by successively removing those that overlap higher-priority ones.

Usage

removeOverlappingRanges(
  x,
  minDist = 7L,
  retIndices = FALSE,
  ignore.strand = FALSE
)

Arguments

x A GRanges, sorted by (decreasing) importance.
minDist Minimum distance between ranges.
retIndices Logical; whether to return the indices of entries to remove, rather than the filtered GRanges.
ignore.strand Logical. Whether the strand of the input ranges should be ignored or not.

Value

A filtered GRanges, or an integer vector of indices to be removed if ‘retIndices==TRUE’.

Examples

library(GenomicRanges)
gr <- GRanges(seqnames=rep("A",4), IRanges(start=c(10,25,45,35), width=6))
removeOverlappingRanges(gr, minDist=7)
### SampleKdModel

**Example KdModel (hsa-miR-155-5p)**

**Description**


**Value**

a ‘KdModel’ object

**Examples**

```r
data(SampleKdModel)
SampleKdModel
```

### SampleTranscript

**Example transcript sequence**

**Description**

An artificial transcript sequence used for examples.

**Value**

a named character vector of length 1.

### viewTargetAlignment

**viewTargetAlignment**

**Description**

viewTargetAlignment
Usage

```r
viewTargetAlignment(
  m,
  miRNA,
  seqs = NULL,
  flagBulgeMatches = FALSE,
  p3.params = list(),
  min3pMatch = 3L,
  hideSingletons = FALSE,
  UGsub = TRUE,
  ...,
  outputType = c("print", "data.frame", "plot", "ggplot")
)
```

Arguments

- `m`: A GRanges of length 1 giving the information for a given match, as produced by `findSeedMatches`.
- `miRNA`: A miRNA sequence, or a `KdModel` object of the miRNA corresponding to the match in `m`; alternatively, a `KdModelList` including the model.
- `seqs`: The sequences corresponding to the seqnames of `m`. Not needed if `m` contains the target sequences.
- `flagBulgeMatches`: Logical; whether to flag matches inside the bulge (default FALSE).
- `p3.params`: See `findSeedMatches`.
- `min3pMatch`: The minimum 3' alignment for any to be plotted.
- `hideSingletons`: Logical; whether to hide isolated single base-pair matches.
- `UGsub`: Logical; whether to show U-G matches.
- `...`: Passed to `text` if `outputType="plot"`.
- `outputType`: Either `"print"` (default, prints to console), `"data.frame"`, or `"plot"`.

Value

Returns nothing `outputType="print"`. If `outputType="data.frame"`, returns a `data.frame` containing the alignment strings; if `outputType="ggplot"` returns a `ggplot` object.

Examples

```r
data(SampleKdModel)
seq <- c(seq1="CGACCCCTATCAGTGCGAGCATTAAAT")
m <- findSeedMatches(seq, SampleKdModel, verbose=FALSE)
viewTargetAlignment(m, miRNA=SampleKdModel, seqs=seq)
```
Index

[,KdModelList,ANY-method (KdModelList-methods), 13
aggregateMatches, 2
assignKdType, 3
c,KdModel-method (KdModel), 12
conservation, 4
data.table, 3
dummyKdData, 5
findSeedMatches, 5, 17
get3pAlignment, 7
get8merRange, 8
getKdModel, 9
getKmers, 9
getMatchTypes, 8, 10
getRandomSeq, 11
getSeed8mers, 11
KdModel, 12, 12, 13, 14, 17
KdModel-class (KdModel), 12
KdModel-methods (KdModel), 12
KdModelList, 5, 12-14, 17
KdModelList (KdModel-class), 13
KdModelList-class, 13
KdModelList-methods, 13
KdModelList-methods,KdModelList-method (KdModelList-methods), 13
plotKdModel, 14
removeOverlappingRanges, 15
SampleKdModel, 16
SampleTranscript, 16
setDTthreads, 3
show,KdModel-method (KdModel), 12
summary,KdModel-method (KdModel), 12
summary,KdModelList-method (KdModelList-methods), 13
viewTargetAlignment, 16

18