

# Package ‘periodicDNA’

December 6, 2024

**Type** Package

**Title** Set of tools to identify periodic occurrences of k-mers in DNA sequences

**Version** 1.16.0

**Date** 2021-11-21

**Encoding** UTF-8

**Description** This R package helps the user identify k-mers (e.g. di- or tri-nucleotides) present periodically in a set of genomic loci (typically regulatory elements). The functions of this package provide a straightforward approach to find periodic occurrences of k-mers in DNA sequences, such as regulatory elements. It is not aimed at identifying motifs separated by a conserved distance; for this type of analysis, please visit MEME website.

**URL** <https://github.com/js2264/periodicDNA>

**BugReports** <https://github.com/js2264/periodicDNA/issues>

**RoxygenNote** 7.1.0

**Depends** R (>= 4.0), Biostrings, GenomicRanges, IRanges, BSgenome, BiocParallel

**Imports** S4Vectors, rtracklayer, stats, GenomeInfoDb, magrittr, zoo, ggplot2, methods, parallel, cowplot

**Suggests** BSgenome.Scerevisiae.UCSC.sacCer3, BSgenome.Celegans.UCSC.ce11, BSgenome.Dmelanogaster.UCSC.dm6, BSgenome.Drerio.UCSC.danRer10, BSgenome.Hsapiens.UCSC.hg38, BSgenome.Mmusculus.UCSC.mm10, reticulate, testthat, covr, knitr, rmarkdown, pkgdown

**VignetteBuilder** knitr

**biocViews** SequenceMatching, MotifDiscovery, MotifAnnotation, Sequencing, Coverage, Alignment, DataImport

**License** GPL-3 + file LICENSE

**git\_url** <https://git.bioconductor.org/packages/periodicDNA>

**git\_branch** RELEASE\_3\_20

**git\_last\_commit** a9cee7d

**git\_last\_commit\_date** 2024-10-29

**Repository** Bioconductor 3.20

**Date/Publication** 2024-12-05

**Author** Jacques Serizay [aut, cre] (<<https://orcid.org/0000-0002-4295-0624>>)

**Maintainer** Jacques Serizay <jacquesserizay@gmail.com>

## Contents

ce11_all_REs . . . . .	2
ce11_ATACseq . . . . .	3
ce11_proms . . . . .	3
ce11_proms_seqs . . . . .	4
ce11_TSSs . . . . .	5
ce11_WW_10bp . . . . .	5
getPeriodicity . . . . .	6
getPeriodicityTrack . . . . .	8
getPeriodicityWithIterations . . . . .	9
plotAggregateCoverage . . . . .	11
plotPeriodicityResults . . . . .	14
setUpBPPARAM . . . . .	15
theme_ggplot2 . . . . .	15

**Index** **18**

---

ce11_all_REs	<i>ce11_all_REs</i>
--------------	---------------------

---

## Description

Regulatory elements annotated in *C. elegans* (ce11) according to Serizay et al. 2020, "Tissue-specific profiling reveals distinctive regulatory architectures for ubiquitous, germline and somatic genes", BiorXiv.

## Usage

```
data(ce11_all_REs)
```

## Format

GRanges

## Source

[BiorXiv](#)

## References

Serizay et al. 2020, "Tissue-specific profiling reveals distinctive regulatory architectures for ubiquitous, germline and somatic genes", BiorXiv. (DOI)

## Examples

```
data(ce11_all_REs)
table(ce11_all_REs$regulatory_class)
table(ce11_all_REs$which.tissues)
```

---

`ce11_ATACseq`*ce11\_ATACseq*

---

**Description**

Sample of ATAC-seq from mixed tissues in *C. elegans* young adults

**Usage**

```
data(ce11_ATACseq)
```

**Format**

RleList

**Source**

[BiorXiv](#)

**References**

Serizay et al. 2020, "Tissue-specific profiling reveals distinctive regulatory architectures for ubiquitous, germline and somatic genes", [BiorXiv](#). (DOI)

**Examples**

```
data(ce11_ATACseq)
ce11_ATACseq
```

---

`ce11_proms`*ce11\_proms*

---

**Description**

Promoters annotated in *C. elegans* (ce11) according to Serizay et al. 2020, "Tissue-specific profiling reveals distinctive regulatory architectures for ubiquitous, germline and somatic genes", [BiorXiv](#).

**Usage**

```
data(ce11_proms)
```

**Format**

GRanges

**Source**

[BiorXiv](#)

## References

Serizay et al. 2020, "Tissue-specific profiling reveals distinctive regulatory architectures for ubiquitous, germline and somatic genes", BiorXiv. ([DOI](#))

## Examples

```
data(ce11_proms)
table(ce11_proms$which.tissues)
```

---

ce11_proms_seqs	<i>ce11_proms_seqs</i>
-----------------	------------------------

---

## Description

Sample of sequences of promoters annotated in *C. elegans* (ce11) according to Serizay et al. 2020, "Tissue-specific profiling reveals distinctive regulatory architectures for ubiquitous, germline and somatic genes", BiorXiv.

## Usage

```
data(ce11_proms_seqs)
```

## Format

DNASTringSet

## Source

[BiorXiv](#)

## References

Serizay et al. 2020, "Tissue-specific profiling reveals distinctive regulatory architectures for ubiquitous, germline and somatic genes", BiorXiv. ([DOI](#))

## Examples

```
data(ce11_proms_seqs)
head(ce11_proms_seqs)
```

---

`ce11_TSSs``ce11_TSSs`

---

**Description**

Coordinates of promoter TSSs annotated in *C. elegans* (ce11) used in Serizay et al. 2020, "Tissue-specific profiling reveals distinctive regulatory architectures for ubiquitous, germline and somatic genes", BiorXiv.

**Usage**

```
data(ce11_TSSs)
```

**Format**

GRanges

**Source**

[BiorXiv](#)

**References**

Serizay et al. 2020, "Tissue-specific profiling reveals distinctive regulatory architectures for ubiquitous, germline and somatic genes", BiorXiv. ([DOI](#))

**Examples**

```
data(ce11_TSSs)
lengths(ce11_TSSs)
ce11_TSSs[[1]]
```

---

`ce11_WW_10bp``ce11_WW_10bp`

---

**Description**

Sample of WW 10-bp periodicity track generated by `getPeriodicityTrack()` in `ce11` over annotated accessible sites, with default parameters

**Usage**

```
data(ce11_WW_10bp)
```

**Format**

RleList

**Source**

[BiorXiv](#)

## References

Serizay et al. 2020, "Tissue-specific profiling reveals distinctive regulatory architectures for ubiquitous, germline and somatic genes", BiorXiv. (DOI)

## Examples

```
data(ce11_WW_10bp)
ce11_WW_10bp
```

---

getPeriodicity	<i>A function to compute k-mer periodicity in sequence(s).</i>
----------------	--

---

## Description

This function takes a set of sequences and a k-mer of interest, map a k-mer of interest in these sequences, computes all the pairwise distances (distogram), normalize it for distance decay, and computes the resulting power spectral density of the normalized distogram.

## Usage

```
getPeriodicity(x, motif, ...)

## S3 method for class 'DNAStrngSet'
getPeriodicity(
  x,
  motif,
  range_spectrum = seq(1, 200),
  BPPARAM = setUpBPPARAM(1),
  roll = 3,
  verbose = TRUE,
  sample = 0,
  n_shuffling = 0,
  cores_shuffling = 1,
  cores_computing = 1,
  order = 1,
  ...
)

## S3 method for class 'GRanges'
getPeriodicity(x, motif, genome = "BSgenome.Celegans.UCSC.ce11", ...)

## S3 method for class 'DNAStrng'
getPeriodicity(x, motif, ...)
```

## Arguments

x	a DNAStrng, DNAStrngSet or GRanges object.
motif	a k-mer of interest
...	Arguments passed to S3 methods

range_spectrum	Numeric vector Range of the distogram to use to run the Fast Fourier Transform on (default: 1:200, i.e. all pairs of k-mers at a maximum of 200 bp from each other).
BPPARAM	split the workload over several processors using BiocParallel
roll	Integer Window to smooth the distribution of pairwise distances (default: 3, to discard the 3-bp periodicity of dinucleotides which can be very strong in vertebrate genomes)
verbose	Boolean
sample	Integer if > 0, will randomly sample this many integers from the dists vector before normalization. This ensures consistency when looking at periodicity in different genomes, since different genomes will have different GC percent
n_shuffling	Integer, how many times should the sequences be shuffled? (default = 0)
cores_shuffling	integer, Number of cores used for shuffling (used if n_shuffling > 0)
cores_computing	integer, split the workload over several processors using BiocParallel (used if n_shuffling > 0)
order	Integer, which order to take into consideration for shuffling (ushuffle python library must be installed for orders > 1) (used if n_shuffling > 0)
genome	genome ID, BSgenome or DNASTringSet object (optional, if x is a GRanges)

### Value

A list containing the results of getPeriodicity function.

- The dists vector is the raw vector of all distances between any possible k-mer.
- The hist.data.frame is the distribution of distances over range\_spectrum.
- The normalized\_hist is the raw hist, normalized for decay over increasing distances.
- The spectra object is the output of the FFT applied over normalized\_hist.
- The PSD data frame is the power spectral density scores over given frequencies.
- The motif object is the k-mer being analysed.
- The final periodicity metrics computed by getPeriodicity()

If getPeriodicity() is ran with n\_shuffling > 0, the resulting list also contains PSD values computed when iterating through shuffled sequences.

### Methods (by class)

- DNASTringSet: S3 method for DNASTringSet
- GRanges: S3 method for GRanges
- DNASTring: S3 method for DNASTring

### Examples

```
data(ce11_proms_seqs)
periodicity_result <- getPeriodicity(
  ce11_proms_seqs[1:100],
  motif = 'TT'
)
```

```

head(periodicity_result$PSD)
plotPeriodicityResults(periodicity_result)
#
data(ce11_TSSs)
periodicity_result <- getPeriodicity(
  ce11_TSSs[['Ubiq. ']][1:10],
  motif = 'TT',
  genome = 'BSgenome.Celegans.UCSC.ce11'
)
head(periodicity_result$PSD)
plotPeriodicityResults(periodicity_result)
#
data(ce11_TSSs)
periodicity_result <- getPeriodicity(
  ce11_TSSs[['Ubiq. ']][1:10],
  motif = 'TT',
  genome = 'BSgenome.Celegans.UCSC.ce11',
  n_shuffling = 10
)
head(periodicity_result$PSD)
plotPeriodicityResults(periodicity_result)

```

---

getPeriodicityTrack     *Function to generate a k-mer periodicity track*

---

## Description

This function takes a set of GRanges in a genome, recover the corresponding sequences and divides them using a sliding window. For each sub-sequence, it then computes the PSD value of a k-mer of interest at a chosen period, and generates a linear .bigWig track from these values.

## Usage

```

getPeriodicityTrack(
  genome = NULL,
  granges,
  motif = "WW",
  period = 10,
  BPPARAM = setUpBPPARAM(1),
  extension = 1000,
  window_size = 100,
  step_size = 2,
  range_spectrum = seq(5, 50),
  smooth_track = 20,
  bw_file = NULL
)

```

## Arguments

genome	DNAStrngSet, BSgenome or genome ID
granges	GRanges object
motif	character, k-mer of interest.



period	Integer, the period of the k-mer to study (default=10).
BPPARAM	split the workload over several processors using BiocParallel
extension	Integer, the width the GRanges are going to be extended to (default 1000).
window_size	Integer, the width of the bins to split the GRanges objects in (default 100).
step_size	Integer, the increment between bins over GRanges (default 2).
range_spectrum	Numeric vector, the distances between nucleotides to take into consideration when performing Fast Fourier Transform (default seq_len(50)).
smooth_track	Integer, smooth the resulting track
bw_file	character, the name of the output bigWig track

**Value**

Rlelist and a bigWig track in the working directory.

**Examples**

```
data(ce11_proms)
track <- getPeriodicityTrack(
  genome = 'BSgenome.Celegans.UCSC.ce11',
  ce11_proms[1],
  extension = 200,
  window_size = 100,
  step_size = 10,
  smooth_track = 1,
  motif = 'WW',
  period = 10,
  BPPARAM = setUpBPPARAM(1)
)
track
unlink(
  'BSgenome.Celegans.UCSC.ce11_WW_10-bp-periodicity_g-100^10_smooth-1.bw'
)
```

---

getPeriodicityWithIterations

*A function to compute PSDs with iterations*

---

**Description**

This function computes PSD values of a given k-mer of interest in a set of input sequences. It also iterates the PSD calculation process over shuffled sequences, if `n_shuffling` is used.

**Usage**

```
getPeriodicityWithIterations(x, ...)

## S3 method for class 'DNAStringSet'
getPeriodicityWithIterations(
  x,
  motif,
```

```

    n_shuffling = 10,
    cores_shuffling = 1,
    cores_computing = 1,
    order = 1,
    verbose = 1,
    ...
)

## S3 method for class 'GRanges'
getPeriodicityWithIterations(x, genome, ...)

```

**Arguments**

<code>x</code>	DNASTringSet, sequences of interest
<code>...</code>	Arguments passed to S3 methods
<code>motif</code>	character, k-mer of interest
<code>n_shuffling</code>	integer, Number of shuffling
<code>cores_shuffling</code>	integer, Number of cores used for shuffling
<code>cores_computing</code>	integer, split the workload over several processors using BiocParallel
<code>order</code>	Integer, which order to take into consideration for shuffling (ushuffle python library must be installed for orders > 1)
<code>verbose</code>	integer, Should the function be verbose?
<code>genome</code>	genome ID, BSgenome or DNASTringSet object (optional, if x is a GRanges)

**Value**

Several metrics

**Methods (by class)**

- DNASTringSet: S3 method for DNASTring
- GRanges: S3 method for GRanges

**Examples**

```

data(ce11_proms_seqs)
res <- getPeriodicityWithIterations(
  ce11_proms_seqs[1:10],
  genome = 'BSgenome.Celegans.UCSC.ce11',
  motif = 'TT',
  cores_shuffling = 1
)
res$observed_PSD
res$shuffled_PSD

```

---

plotAggregateCoverage *A function to plot aggregated signals over sets of GRanges*

---

## Description

This function takes one or several RleList genomic tracks (e.g. imported by rtraklayer::import(..., as = 'Rle')) and one or several GRanges objects. It computes coverage of the GRanges by the genomic tracks and returns an aggregate coverage plot.

## Usage

```
plotAggregateCoverage(x, ...)

## S3 method for class 'CompressedRleList'
plotAggregateCoverage(x, granges, ...)

## S3 method for class 'SimpleRleList'
plotAggregateCoverage(
  x,
  granges,
  colors = NULL,
  xlab = "Center of elements",
  ylab = "Score",
  xlim = NULL,
  ylim = NULL,
  quartiles = c(0.025, 0.975),
  verbose = FALSE,
  bin = 1,
  plot_central = TRUE,
  run_in_parallel = FALSE,
  split_by_granges = FALSE,
  norm = "none",
  ...
)

## S3 method for class 'list'
plotAggregateCoverage(
  x,
  granges,
  colors = NULL,
  xlab = "Center of elements",
  ylab = "Score",
  xlim = NULL,
  ylim = NULL,
  quartiles = c(0.025, 0.975),
  verbose = FALSE,
  bin = 1,
  plot_central = TRUE,
  split_by_granges = TRUE,
  split_by_track = FALSE,
  free_scales = FALSE,
```

```

run_in_parallel = FALSE,
norm = "none",
...
)

```

### Arguments

x	a single signal track (CompressedRleList or SimpleRleList class), or several signal tracks (SimpleRleList or CompressedRleList class) grouped in a named list
...	additional parameters
granges	a GRanges object or a named list of GRanges
colors	a vector of colors
xlab	x axis label
ylab	y axis label
xlim	y axis limits
ylim	y axis limits
quartiles	Which quantiles to use to determine y scale automatically?
verbose	Boolean
bin	Integer Width of the window to use to smooth values by zoo::rollMean
plot_central	Boolean Draw a vertical line at 0
run_in_parallel	Boolean Should the plots be computed in parallel using mclapply?
split_by_granges	Boolean Facet plots over the sets of GRanges
norm	character Should the signal be normalized ('none', 'zscore' or 'log2')?
split_by_track	Boolean Facet plots by the sets of signal tracks
free_scales	Boolean Should each facet have independent y-axis scales?

### Value

An aggregate coverage plot.

### Methods (by class)

- CompressedRleList: S3 method for CompressedRleList
- SimpleRleList: S3 method for SimpleRleList
- list: S3 method for list

### Examples

```

data(ce11_ATACseq)
data(ce11_WW_10bp)
data(ce11_proms)

p1 <- plotAggregateCoverage(
  ce11_ATACseq,
  resize(ce11_proms[1:100], fix = 'center', width = 1000)
)

```

```

p1

proms <- resize(ce11_proms[1:100], fix = 'center', width = 400)
p2 <- plotAggregateCoverage(
  ce11_ATACseq,
  list(
    'Ubiq & Germline promoters' =
      proms[proms$which.tissues %in% c('Ubiq.', 'Germline')],
    'Other promoters' =
      proms[!(proms$which.tissues %in% c('Ubiq.', 'Germline'))]
  )
)
p2

p3 <- plotAggregateCoverage(
  list(
    'atac' = ce11_ATACseq,
    'WW_10bp' = ce11_WW_10bp
  ),
  proms,
  norm = 'zscore'
)
p3

p4 <- plotAggregateCoverage(
  list(
    'ATAC-seq' = ce11_ATACseq,
    'WW 10-bp periodicity' = ce11_WW_10bp
  ),
  list(
    'Ubiq & Germline promoters' =
      proms[proms$which.tissues %in% c('Ubiq.', 'Germline')],
    'Other promoters' =
      proms[!(proms$which.tissues %in% c('Ubiq.', 'Germline'))]
  ),
  norm = 'zscore'
)
p4

p5 <- plotAggregateCoverage(
  list(
    'ATAC-seq' = ce11_ATACseq,
    'WW 10-bp periodicity' = ce11_WW_10bp
  ),
  list(
    'Ubiq & Germline promoters' =
      proms[proms$which.tissues %in% c('Ubiq.', 'Germline')],
    'Other promoters' =
      proms[!(proms$which.tissues %in% c('Ubiq.', 'Germline'))]
  ),
  split_by_granges = FALSE,
  split_by_track = TRUE,
  norm = 'zscore'
)
p5

```

---

plotPeriodicityResults

*Plot the output of getPeriodicity()*

---

### Description

This function plots some results from the result of `getPeriodicity()`. It plots the raw histogram, the distance-decay normalized histogram and the resulting PSD values. If a shuffled control has been performed by `getPeriodicity()`, it also displays it.

### Usage

```
plotPeriodicityResults(
  results,
  periods = c(2, 20),
  filter_periods = TRUE,
  facet_control = TRUE,
  xlim = NULL,
  fdr_threshold = 0.05,
  ...
)
```

### Arguments

<code>results</code>	The output of <code>getPeriodicity</code> function.
<code>periods</code>	Vector a numerical vector of length 2, to specify the x-axis limits
<code>filter_periods</code>	Boolean Should the x-axis be constrained to the periods?
<code>facet_control</code>	Boolean should the shuffling plots be faceted?
<code>xlim</code>	Integer x axis upper limit in raw and norm. histograms
<code>fdr_threshold</code>	Float, significance threshold
<code>...</code>	Additional theme arguments passed to <code>theme_ggplot2()</code>

### Value

list A list containing four ggplots

### Examples

```
data(ce11_TSSs)
periodicity_result <- getPeriodicity(
  ce11_TSSs[['Ubiq.']] [1:100],
  genome = 'BSgenome.Celegans.UCSC.ce11',
  motif = 'TT',
  BPPARAM = setUpBPPARAM(1)
)
head(periodicity_result$PSD)
plotPeriodicityResults(periodicity_result)
plotPeriodicityResults(periodicity_result, xlim = 150)
plotPeriodicityResults(
  periodicity_result, xlim = 150, filter_periods = FALSE
```

```

)
plotPeriodicityResults(
  periodicity_result, xlim = 150, facet_control = FALSE
)

```

---

setUpBPPARAM

*setUpBPPARAM*


---

### Description

A function to dynamically select MulticoreParam or SnowParam (if Windows)

### Usage

```
setUpBPPARAM(nproc = 1)
```

### Arguments

nproc                    number of processors

### Value

A BPPARAM object

### Examples

```
BPPARAM <- setUpBPPARAM(1)
```

---

theme\_ggplot2

*Personal ggplot2 theming function, adapted from roboto-condensed at <https://github.com/hrbrmstr/hrbrthemes/>*


---

### Description

Personal ggplot2 theming function, adapted from roboto-condensed at <https://github.com/hrbrmstr/hrbrthemes/>

### Usage

```

theme_ggplot2(
  grid = TRUE,
  border = TRUE,
  base_size = 8,
  plot_title_size = 12,
  plot_title_face = "plain",
  plot_title_margin = 5,
  subtitle_size = 11,
  subtitle_face = "plain",
  subtitle_margin = 5,
  strip_text_size = 10,
  strip_text_face = "bold",

```

```

caption_size = 9,
caption_face = "plain",
caption_margin = 3,
axis_text_size = base_size,
axis_title_size = 9,
axis_title_face = "plain",
axis_title_just = "rt",
panel_spacing = grid::unit(2, "lines"),
grid_col = "#cccccc",
plot_margin = margin(12, 12, 12, 12),
axis_col = "#cccccc",
axis = FALSE,
ticks = FALSE
)

```

### Arguments

grid	panel grid ('TRUE', 'FALSE', or a combination of 'X', 'x', 'Y', 'y')
border	border if 'TRUE' add border
base_size	base font size
plot_title_size, plot_title_margin	plot title size and margin
plot_title_face	plot title face
subtitle_face, subtitle_size	plot subtitle face and size
subtitle_margin	plot subtitle margin bottom (single numeric value)
strip_text_face, strip_text_size	facet label font face and size
caption_face, caption_size, caption_margin	plot caption face, size and margin
axis_text_size	font size of axis text
axis_title_face, axis_title_size	axis title font face and size
axis_title_just	axis title font justification one of '[blmcr]'
panel_spacing	panel spacing (use 'unit()')
grid_col	grid color
plot_margin	plot margin (specify with [ggplot2::margin])
axis_col	axis color
axis	add x or y axes? 'TRUE', 'FALSE', "'xy'"
ticks	ticks if 'TRUE' add ticks

### Value

theme A ggplot theme



**Examples**

```
library(ggplot2)

ggplot(mtcars, aes(mpg, wt)) +
  geom_point() +
  labs(x="Fuel efficiency (mpg)", y="Weight (tons)",
       title="Seminal ggplot2 scatterplot example") +
  theme_ggplot2()
```

# Index

## \* datasets

- ce11\_all\_REs, [2](#)
- ce11\_ATACseq, [3](#)
- ce11\_proms, [3](#)
- ce11\_proms\_seqs, [4](#)
- ce11\_TSSs, [5](#)
- ce11\_WW\_10bp, [5](#)

- ce11\_all\_REs, [2](#)
- ce11\_ATACseq, [3](#)
- ce11\_proms, [3](#)
- ce11\_proms\_seqs, [4](#)
- ce11\_TSSs, [5](#)
- ce11\_WW\_10bp, [5](#)

- getPeriodicity, [6](#)
- getPeriodicityTrack, [8](#)
- getPeriodicityWithIterations, [9](#)

- plotAggregateCoverage, [11](#)
- plotPeriodicityResults, [14](#)

- setUpBPPARAM, [15](#)

- theme\_ggplot2, [15](#)