

# Package ‘segmenter’

October 11, 2022

**Type** Package

**Title** Perform Chromatin Segmentation Analysis in R by Calling ChromHMM

**Version** 1.2.0

**Description** Chromatin segmentation analysis transforms ChIP-seq data into signals over the genome. The latter represents the observed states in a multivariate Markov model to predict the chromatin's underlying states. ChromHMM, written in Java, integrates histone modification datasets to learn the chromatin states de-novo. The goal of this package is to call chromHMM from within R, capture the output files in an S4 object and interface to other relevant Bioconductor analysis tools. In addition, segmenter provides functions to test, select and visualize the output of the segmentation.

**License** GPL-3

**Encoding** UTF-8

**RoxygenNote** 7.1.1

**Depends** R (>= 4.1)

**BugReports** <https://github.com/MahShaaban/segmenter/issues>

**Suggests** testthat, knitr, rmarkdown,  
TxDb.Hsapiens.UCSC.hg18.knownGene, Gviz

**Imports** ChIPseeker, GenomicRanges, SummarizedExperiment, IRanges,  
S4Vectors, bamsignals, ComplexHeatmap, graphics, stats, utils,  
methods, chromhmmData

**VignetteBuilder** knitr

**biocViews** Software, HistoneModification

**LazyData** True

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

**git\_branch** RELEASE\_3\_15

**git\_last\_commit** f9cf94d

**git\_last\_commit\_date** 2022-04-26

**Date/Publication** 2022-10-11

**Author** Mahmoud Ahmed [aut, cre] (<<https://orcid.org/0000-0002-4377-6541>>)

**Maintainer** Mahmoud Ahmed <mahmoud.s.fahmy@students.kasralainy.edu.eg>

**R topics documented:**

.Binarize . . . . .	3
.LearnModel . . . . .	3
accessors . . . . .	4
annotate_segments . . . . .	7
binarize_bam . . . . .	8
binarize_bed . . . . .	9
compare_models . . . . .	10
count_reads_ranges . . . . .	10
emissions_file . . . . .	11
enrichment_files . . . . .	11
get_frequency . . . . .	12
get_width . . . . .	13
learn_model . . . . .	13
merge_segments_bins . . . . .	15
methods . . . . .	16
model_file . . . . .	16
overlap_files . . . . .	17
plot_heatmap . . . . .	18
range_bins . . . . .	18
range_counts . . . . .	19
read_bam_file . . . . .	20
read_bins_file . . . . .	20
read_cellmark_file . . . . .	21
read_chromsize_file . . . . .	22
read_emissions_file . . . . .	22
read_enrichment_file . . . . .	23
read_model_file . . . . .	24
read_overlap_file . . . . .	24
read_segements_file . . . . .	25
read_transitions_file . . . . .	26
segmentation . . . . .	26
segments_files . . . . .	27
test_obj . . . . .	28
test_objs . . . . .	28
tidy_ranges . . . . .	29
transitions_file . . . . .	29

---

.Binarize                      *Call Java BinarizeBed*

---

**Description**

Call the Java module BinarizeBed which binarize a bed file of the aligned reads.

**Usage**

.Binarize(inputdir, cellmarkfiletable, chromsizefile, binsize, outputdir, type)

**Arguments**

inputdir                      A string. The path to bed files.  
cellmarkfiletable            A tab delimited files of three columns. The columns contains the cell, mark and the name or the bed file.  
chromsizefile                A string. The path to the chromosomes sizes file.  
binsize                        An integer. The bin size to use. Default is 200.  
outputdir                     A string. The path to a directory where output will be written.  
type                            A string. The file type 'bam' or 'bed'.

**Value**

NULL. Output files are written to the output directory.

**See Also**

binarize\_bed

---

.LearnModel                    *Call Java LearnModel*

---

**Description**

Call the Java module LearnModel which learns a multi-state model from ChIP-seq data.

**Usage**

```
.LearnModel(
  inputdir,
  outputdir,
  numstates,
  coordsdir,
  anchorsdir,
  chromsizefile,
  assembly,
  optional
)
```

**Arguments**

inputdir	A string. The path to binarized files.
outputdir	A string. The path to a directory where output will be written.
numstates	An integer. The number of desired states in the model.
coordsdir	A string. The path to genomic coordinates files.
anchorsdir	A string. The path to the genomic anchors files.
chromsizefile	A string. The path to the chromosomes sizes file.
assembly	A string. The name of the genomic assembly.
optional	A string. Other optional arguments passed to the Java command.

**Value**

NULL. Output files are written to the output directory.

**See Also**

learn\_model

---

accessors

*Accessors for the segmentation objects*

---

**Description**

These functions can be used to access the contents of segmentation objects as well as modifying them.

**Usage**

```
model(object)

## S4 method for signature 'segmentation'
model(object)

emission(object)

## S4 method for signature 'segmentation'
emission(object)

transition(object)

## S4 method for signature 'segmentation'
transition(object)

overlap(object, ...)

## S4 method for signature 'segmentation'
overlap(object, cell)

TSS(object, ...)

## S4 method for signature 'segmentation'
TSS(object, cell)

TES(object, ...)

## S4 method for signature 'segmentation'
TES(object, cell)

segment(object, ...)

## S4 method for signature 'segmentation'
segment(object, cell)

bins(object, ...)

## S4 method for signature 'segmentation'
bins(object, cell)

counts(object, ...)

## S4 method for signature 'segmentation'
counts(object, cell)

likelihood(object)
```

```
## S4 method for signature 'segmentation'
likelihood(object)

cells(object)

## S4 method for signature 'segmentation'
cells(object)

states(object)

## S4 method for signature 'segmentation'
states(object)

markers(object)

## S4 method for signature 'segmentation'
markers(object)
```

### Arguments

object	An object of class segmentation
...	Other argument passed to the accessors
cell	A string

### Value

The data in the corresponding slot or a subset of it.

### See Also

segmentation

### Examples

```
model(test_obj)

emission(test_obj)

transition(test_obj)

overlap(test_obj)
overlap(test_obj, cell = 'K562')

TSS(test_obj)
TSS(test_obj, cell = 'K562')

TES(test_obj)
TES(test_obj, cell = 'K562')

segment(test_obj)
```

```

segment(test_obj, cell = 'K562')

bins(test_obj)

counts(test_obj)

likelihood(test_obj)

cells(test_obj)

states(test_obj)

markers(test_obj)

```

---

annotate\_segments      *Annotate segments*

---

### Description

Annotate the GRanges objects of the segments using [annotatePeak](#) (see for details)

### Usage

```
annotate_segments(segments, ...)
```

### Arguments

segments	A GRanges object. Usually the output of calling <code>segment</code> on the the output object of <code>lean_model</code> .
...	Other arguments passed to <a href="#">annotatePeak</a>

### Value

A GRanges object which is identical to the input in addition to the annotations as metadata columns.

### Examples

```

library(TxDb.Hsapiens.UCSC.hg18.knownGene)
txdb <- TxDb.Hsapiens.UCSC.hg18.knownGene
segs <- segment(test_obj)
segs_annotated <- annotate_segments(segs, TxDb = txdb, verbose = FALSE)

```

---

binarize\_bam                      *Binarize the bam files*

---

## Description

Transform the aligned reads into a binary format.

## Usage

```
binarize_bam(  
  inputdir,  
  cellmarkfiletable,  
  chromsizefile,  
  binsize = 200,  
  outputdir  
)
```

## Arguments

inputdir	A string. The directory of the bam files.
cellmarkfiletable	A string. The path to the input files table. Only
chromsizefile	A string. The path to the chromosomes sizes file.
binsize	An integer. The number in bp used to generate binarized files.
outputdir	A string. The path to a directory where output will be written.

## Value

NULL. Write files to the outputdir

## See Also

Binarize binarize\_bed

## Examples

```
# locate input and output files  
inputdir <- system.file("extdata", package = "bamsignals")  
cellmarkfiletable <- system.file('extdata',  
                                 'cell_mark_table.tsv',  
                                 package = 'segmenter')  
chromsizefile <- system.file('extdata/CHROMSIZES',  
                             'hg18.txt',  
                             package = 'chromhmmData')  
  
outputdir <- tempdir()  
  
# run command
```



```
binarize_bam(inputdir,  
             chromsizefile = chromsizefile,  
             cellmarkfiletable = cellmarkfiletable,  
             outputdir = outputdir)  
  
# show output files  
list.files(outputdir, pattern = '*_binary.txt')
```

---

binarize\_bed

*Binarize the bed files*

---

## Description

Transform the aligned reads into a binary format.

## Usage

```
binarize_bed(  
  inputdir,  
  cellmarkfiletable,  
  chromsizefile,  
  binsize = 200,  
  outputdir  
)
```

## Arguments

inputdir	A string. The directory of the bam files.
cellmarkfiletable	A string. The path to the input files table. Only
chromsizefile	A string. The path to the chromosomes sizes file.
binsize	An integer. The number in bp used to generate binarized files.
outputdir	A string. The path to a directory where output will be written.

## Value

NULL. Write files to the outputdir

## See Also

Binarize binarize\_bam

---

compare_models	<i>Compare two or more models</i>
----------------	-----------------------------------

---

**Description**

Compare two or more models

**Usage**

```
compare_models(objs, type = "emission", plot = FALSE, ...)
```

**Arguments**

objs	A list of segmentation items
type	A string. What to compare. Default to 'emission'
plot	A logical.
...	Other arguments passed to plot

**Value**

A numeric vector or a plot with the same values.

**Examples**

```
compare_models(test_objs)
compare_models(test_objs, type = 'likelihood')
```

---

count_reads_ranges	<i>Count reads in GRanges objects from bam files</i>
--------------------	--

---

**Description**

Count reads in GRanges objects from bam files

**Usage**

```
count_reads_ranges(ranges, cellmarkfiletable, inputbamdir)
```

**Arguments**

ranges	A GRanges to count in.
cellmarkfiletable	A string. The path to the input files table.
inputbamdir	A string. The path to the input bam files directory.

**Value**

A SummarizedExperiment object with ranges as its rowRanges and the counts as the assay.

---

emissions_file	<i>Make emissions file name</i>
----------------	---------------------------------

---

**Description**

Make emissions file name

**Usage**

```
emissions_file(numstates)
```

**Arguments**

numstates      An integer

**Value**

A string

**Examples**

```
emissions_file(3)
```

---

enrichment_files	<i>Make enrichment file names</i>
------------------	-----------------------------------

---

**Description**

Make enrichment file names

**Usage**

```
enrichment_files(numstates, cells, table = "RefSeq", annotation = "TSS")
```

**Arguments**

numstates      An integer  
 cells            A character vector  
 table            A string  
 annotation      A string

**Value**

A character vector

**Examples**

```
enrichment_files(3, 'K562')
```

---

`get_frequency`*Get the frequency of the segments in each cell type*

---

**Description**

Get the frequency of the segments in each cell type

**Usage**

```
get_frequency(segments, normalize = FALSE, tidy = FALSE, plot = FALSE, ...)
```

**Arguments**

<code>segments</code>	A GRanges object. Usually the output of calling <code>segment</code> on the the output object of <code>lean_model</code> .
<code>normalize</code>	A logical. Whether the frequency should be normalized by the total number of segments
<code>tidy</code>	A logical.
<code>plot</code>	A logical.
<code>...</code>	Other arguments passed to <code>barplot</code>

**Value**

A data.frame when `tidy` is TRUE otherwise a matrix or a plot

**Examples**

```
get_frequency(segment(test_obj))  
get_frequency(segment(test_obj), normalize = TRUE)
```

---

get_width	<i>Get the width of the segments in each cell type</i>
-----------	--

---

**Description**

Get the width of the segments in each cell type

**Usage**

```
get_width(segments, average = FALSE)
```

**Arguments**

segments	A GRanges object. Usually the output of calling segment on the the output object of learn_model.
average	A logical. Whether the width should be averaged across cells.

**Value**

A data.frame

**Examples**

```
get_width(segment(test_obj))  
get_width(segment(test_obj), average = TRUE)
```

---

learn_model	<i>Learn a multi-state model from chromatin data</i>
-------------	--

---

**Description**

Integrate multiple ChIP-seq chromatin datasets of histone modifications, transcription factors or other DNA binding proteins to build a multi-state model of the combinatorial and spatial frequently occurring patterns. The function uses as an input binarized ChIP-seq data and the genome annotations on which the states will be discovered.

**Usage**

```
learn_model(  
  inputdir,  
  outputdir,  
  numstates,  
  coordsdir,  
  anchorsdir,  
  chromsizefile,
```

```

assembly,
cells,
annotation,
binsize,
inputbamdir,
cellmarkfiletable,
read_only = FALSE,
read_bins = FALSE,
counts = FALSE
)

```

### Arguments

inputdir	A string. The path to binarized files.
outputdir	A string. The path to a directory where output will be written.
numstates	An integer. The number of desired states in the model.
coordsdir	A string. The path to genomic coordinates files.
anchorsdir	A string. The path to the genomic anchors files.
chromsizefile	A string. The path to the chromosomes sizes file.
assembly	A string. The name of the genomic assembly.
cells	A character vector. The names of the cells as they occur in the binarized files (first line).
annotation	A string. The name of the type of annotation as it occurs in the genomic annotation files.
binsize	An integer. The number in bp used to generate binarized files.
inputbamdir	A string. The path to the input bam files. Only used when count = TRUE.
cellmarkfiletable	A string. The path to the input files table. Only used when bins = TRUE.
read_only	A logical. Default is FALSE. Whether to look for and load output files or generate the model from scratch.
read_bins	A logical. Default is FALSE. Whether to load the binarized data into the output object.
counts	A logical. Default is FALSE. Whether to load the reads counts in bins data into the output object.

### Details

By default, this functions runs the analysis commands, writes the output to files and loads it into an object of class [segmentation](#). In addition, the binarized data and the reads counts in the bins can be loaded. When `read_only` is TRUE. The functions looks for previously generated files in the output directory and load them without rerunning the commands.

### Value

An object of class [segmentation](#) (see for details) and the files written to the output directory.

**See Also**

LearnModel

**Examples**

```
# locate input and output files
inputdir <- system.file('extdata/SAMPLEDATA_HG18',
                        package = 'segmenter')
outputdir <- tempdir()
coordsdir <- system.file('extdata/COORDS',
                         package = 'chromhmmData')
anchorsdir <- system.file('extdata/ANCHORFILES',
                          package = 'chromhmmData')
chromsizefile <- system.file('extdata/CHROMSIZES',
                             'hg18.txt',
                             package = 'chromhmmData')

# run command
obj <- learn_model(inputdir = inputdir,
                  outputdir = outputdir,
                  coordsdir = coordsdir,
                  anchorsdir = anchorsdir,
                  chromsizefile = chromsizefile,
                  numstates = 3,
                  assembly = 'hg18',
                  cells = c('K562', 'GM12878'),
                  annotation = 'RefSeq',
                  binsize = 200)

# show the output
obj
```

---

merge\_segments\_bins    *Merge segments and bins objects*

---

**Description**

Merge segments and bins objects

**Usage**

```
merge_segments_bins(segments, bins)
```

**Arguments**

segments	A GRanges object. Usually the output of calling segment on the the output object of lean_model.
bins	A SummarizedExperiment object. Usually the output of calling bins on the the output object of lean_model.

**Value**

A SummarizedExperiment object with the segment assignment added to the metadata of the rowRanges.

---

methods	<i>Methods to interact with segmentation objects</i>
---------	--

---

**Description**

These functions can be used to interact with segmentation objects for purposes other than accessing or modifying their contents.

**Usage**

```
## S4 method for signature 'segmentation'
show(object)
```

**Arguments**

object            An object of class segmentation

**Value**

Prints a summary of the segmentation object contents.

**See Also**

segmentation  
accessors

**Examples**

```
show(test_obj)
```

---

model_file	<i>Make model file name</i>
------------	-----------------------------

---

**Description**

Make model file name

**Usage**

```
model_file(numstates)
```



**Arguments**

numstates      An integer

**Value**

A string

**Examples**

```
model_file(3)
```

---

*overlap\_files*      *Make overlap file names*

---

**Description**

Make overlap file names

**Usage**

```
overlap_files(numstates, cells)
```

**Arguments**

numstates      An integer

cells            A character vector

**Value**

A character vector

**Examples**

```
overlap_files(3, 'K562')
```

---

plot_heatmap	<i>Visualize the model output</i>
--------------	-----------------------------------

---

**Description**

Visualize the model output

**Usage**

```
plot_heatmap(obj, type = "emission", ...)
```

**Arguments**

obj	A segmentation object
type	A string. Which kind of parameter to print. Default is 'emission' and possible values are 'emission', 'transition', 'overlap', 'TSS' or 'TES'
...	Other arguments to path to Heatmap

**Value**

A heatmap

**Examples**

```
plot_heatmap(test_obj)
```

---

range_bins	<i>Format the loaded binarized data</i>
------------	---

---

**Description**

The function takes the `data.frames` of the loaded binarized data files and format them into `GRanges` or `SummarizedExperiment` objects.

**Usage**

```
range_bins(bins, chromsizefile, binsize, return = "GRanges", tidy = TRUE)
```

**Arguments**

bins	A list of the <code>read_bins_file</code> output.
chromsizefile	A string. The path to the chromosomes sizes file.
binsize	An integer. The number in bp used to generate binarized files.
return	A string. Possible values are <code>GRanges</code> (default) or <code>SummarizedExperiment</code> .
tidy	A logical. Default is <code>TRUE</code> . Whether to tidy the metadata columns of the <code>GRanges</code> object.

**Value**

GRanges (default) or SummarizedExperiment.

---

 range\_counts
 

---



---

*Format the loaded counts data*


---

**Description**

The function takes the data.frames of the loaded counts data and format them into GRanges or SummarizedExperiment objects.

**Usage**

```
range_counts(
  counts,
  features,
  return = "GRanges",
  tidy = FALSE,
  average = FALSE,
  marks
)
```

**Arguments**

counts	A matrix of the read_bam_file output.
features	A GRanges. That was used to count the bam files.
return	A string. Possible values are GRanges (default) or SummarizedExperiment.
tidy	A logical. Default is TRUE. Whether to tidy the metadata columns of the GRanges object.
average	A logical. Default is FALSE. Whether to average the counts by marks before building the object.
marks	A character vector. The length should equal the number of columns in counts and is used for averaging and renaming the matrix columns.

**Value**

GRanges (default) or SummarizedExperiment.

---

read_bam_file	<i>Read bam files</i>
---------------	-----------------------

---

**Description**

Count the reads in each range of the GRanges object

**Usage**

```
read_bam_file(file, features, ...)
```

**Arguments**

file	A string. The path to the file.
features	A GRanges object.
...	Other arguments passed to <a href="#">bamCount</a> .

**Value**

A matrix

**Examples**

```
# locate the bam file
bam_file <- system.file("extdata", "randomBam.bam", package = "bamsignals")

# load a granges object
rand_anno <- system.file("extdata",
                        "randomAnnot.Rdata",
                        package = "bamsignals")
features <- GenomicRanges::promoters(get(load(rand_anno)))

# count reads in ranges
read_bam_file(bam_file, features)
```

---

read_bins_file	<i>Read bins files</i>
----------------	------------------------

---

**Description**

The files contain the cell and the chromosome info in the first line and the binarized data from all marks in the rest.

**Usage**

```
read_bins_file(file)
```

**Arguments**

file                    A string. The path to the file.

**Value**

A list of 3 items: cell, seqname and binaries.

**Examples**

```
# locate the file
fl <- system.file('extdata/SAMPLEDATA_HG18/',
                  'GM12878_chr11_binary.txt.gz',
                  package = 'segmenter')

# read the file
read_bins_file(fl)
```

---

read\_cellmark\_file     *Read cellmarktable file*

---

**Description**

The file should contain at least three columns: cell, mark and file for the names of the cells/conditions, the available marks and binarized data files.

**Usage**

```
read_cellmark_file(file)
```

**Arguments**

file                    A string. The path to the file.

**Value**

A data.frame

**Examples**

```
# locate the file
fl <- system.file('extdata',
                  'cell_mark_table.tsv',
                  package = 'segmenter')

# read the file
read_cellmark_file(fl)
```

---

read\_chromsize\_file    *Read chromsizefile*

---

### Description

The file should contain exactly two columns. One for the name of the chromosome and the other for its length.

### Usage

```
read_chromsize_file(file)
```

### Arguments

file                    A string. The path to the file.

### Value

A data.frame

### Examples

```
# locate the file
chromsizefile <- system.file('extdata/CHROMSIZES',
                             'hg18.txt',
                             package = 'chromhmmData')

# read the file
read_chromsize_file(chromsizefile)
```

---

read\_emissions\_file    *Read emissions file*

---

### Description

The segments files are the output of running learn\_model and named emissions\_3\_segment.bed

### Usage

```
read_emissions_file(file, states, marks)
```

### Arguments

file                    A string. The path to the file.  
states                   A character vector. The names of the states.  
marks                    A character vector. The names of the marks

**Value**

A matrix

**Examples**

```
# locate the file
fl <- file.path(tempdir(), 'emissions_3.txt')

# read the file
read_emissions_file(fl)
```

---

read\_enrichment\_file *Read enrichment files*

---

**Description**

The segments files are the output of running learn\_model and named <cell>\_3\_TSS.txt or <cell>\_3\_TES.txt.

**Usage**

```
read_enrichment_file(file, states, regions)
```

**Arguments**

file	A string. The path to the file.
states	A character vector. The names of the states.
regions	A character vector. The names of the regions.

**Value**

A matrix

**Examples**

```
# locate the file
fl <- file.path(tempdir(), 'GM12878_3_RefSeqTSS_neighborhood.txt')

# read the file
read_enrichment_file(fl)
```

---

read_model_file	<i>Read modelfile</i>
-----------------	-----------------------

---

**Description**

The model file is the output of running `learn_model` and named `model_#.txt`

**Usage**

```
read_model_file(file)
```

**Arguments**

`file` A string. The path to the file.

**Value**

A `data.frame`

**Examples**

```
# locate the file
modelfile <- file.path(tempdir(), 'model_3.txt')

# read the file
read_model_file(modelfile)
```

---

read_overlap_file	<i>Read segments files</i>
-------------------	----------------------------

---

**Description**

The segments files are the output of running `learn_model` and named `<cell>_3_overlap.txt`

**Usage**

```
read_overlap_file(file, states, regions)
```

**Arguments**

`file` A string. The path to the file.  
`states` A character vector. The names of the states.  
`regions` A character vector. The names of the regions.



**Value**

A matrix

**Examples**

```
# locate the file
fl <- file.path(tempdir(), 'GM12878_3_overlap.txt')

# read the file
read_overlap_file(fl)
```

---

read\_segements\_file    *Read segments files*

---

**Description**

The segments files are the output of running learn\_model and named <cell>\_3\_segment.bed

**Usage**

```
read_segements_file(file, states)
```

**Arguments**

file                    A string. The path to the file.  
states                  A character vector. The names of the states.

**Value**

A data.frame

**Examples**

```
# locate the file
segmentfile <- file.path(tempdir(), 'GM12878_3_segments.bed')

# read the file
segs <- read_segements_file(segmentfile)
head(segs)
```

---

read\_transitions\_file *Read transitions file*

---

### Description

The segments files are the output of running `learn_model` and named `transitions_3_segment.bed`

### Usage

```
read_transitions_file(file, states)
```

### Arguments

<code>file</code>	A string. The path to the file.
<code>states</code>	A character vector. The names of the states.

### Value

A matrix

### Examples

```
# locate the file
fl <- file.path(tempdir(), 'transitions_3.txt')

# read the file
read_transitions_file(fl)
```

---

segmentation *segmentation objects*

---

### Description

The `segmentation` class consists of matrices and lists. The components contain the output of the chromatin segmentation analysis. Loading the input data is optional. The object is returned as a result of calling `learn_model` or reading its already existing output.

### Slots

`model` list. The list consists of 6 items corresponding to the contents of the `model_#.txt` file. These are `number_states` and `number_marks` for the numbers of states and marks in the model; `likelihood` and `probinit` for the likelihood and the initial probabilities of the multi-state model; `transitionprobs` and `emissionprobs` for the probabilities of the transitions and emissions parameters of the model. Can be accessed using `model`.

- emission matrix. The matrix contains the emission parameters of  $n$  states (rows) for  $n$  marks (columns) corresponding to the contents of the `emission_#.txt` file. Can be accessed using [emission](#).
- transition matrix. The matrix contains the transition parameters of  $n$  by  $n$  states corresponding to the contents of the `transition_#.txt` file. Can be accessed using [transition](#).
- overlap list. A list of  $n$  number of cells/conditions items. Each item is a matrix of the overlap enrichment of  $n$  states (rows) at  $n$  genomic annotations (columns) corresponding to the contents of the `<cell>_#_overlap.txt` files. Can be accessed using [overlap](#).
- TSS list. A list of  $n$  number of cells/conditions items. Each item is a matrix of the overlap enrichment of  $n$  states (rows) at  $n$  locations around the transcription start site (TSS) (columns) corresponding to the contents of the `<cell>_#_TSS_neighborhood.txt` files. Can be accessed using [TSS](#).
- TES list. A list of  $n$  number of cells/conditions items. Each item is a matrix of the overlap enrichment of  $n$  states (rows) at  $n$  locations around the transcription end site (TES) (columns) corresponding to the contents of the `<cell>_#_TES_neighborhood.txt` files. Can be accessed using [TES](#).
- segment list. A list of  $n$  number of cells/conditions items. Each item is a [GRanges](#) object containing the segmentation and assigned states as a metadata column 'state'. These contents correspond to the `<cell>_#_segment.bed` files. Annotations of the ranges are optional. Can be accessed using [segment](#).
- bins list. A list of  $n$  number of cells/conditions items. Each item is a [SummarizedExperiment](#) object containing the binarized input data. The coordinates of the bins are saved as the [rowRanges](#) each assigned to a state and the binary data itself is saved as [assay](#). Can be accessed using [bins](#).
- counts list. A list of  $n$  number of cells/conditions items. Each item is a [SummarizedExperiment](#) object containing the read counts in bins. The coordinates of the bins are saved as the [rowRanges](#) each assigned to a state and the counts data itself is saved as [assay](#). Can be accessed using [counts](#).

### See Also

[learn\\_model](#)

---

segments\_files

*Make segments file names*

---

### Description

Make segments file names

### Usage

```
segments_files(numstates, cells)
```

**Arguments**

numstates      An integer  
 cells            A character vector

**Value**

A character vector

**Examples**

```
segments_files(3, 'K562')
```

---

test_obj	<i>A segmentation object generated from the test data</i>
----------	---

---

**Description**

A segmentation object generated by running `lean_model` on the test dataset in `'inst/extdata/ChromHMM/SAMPLEDATA_H'`  
 The source code to this run is in `'inst/script/test_obj.R'`

**Usage**

```
test_obj
```

**Format**

An object of class `segmentation` of length 1.

---

test_objs	<i>A a list of segmentation objects generated from the test data</i>
-----------	--

---

**Description**

A segmentation object generated by running `lean_model` on the test dataset in `'inst/extdata/ChromHMM/SAMPLEDATA_H'`  
 for 3 to 8 states. The source code to this run is in `'inst/script/test_objs.R'`

**Usage**

```
test_objs
```

**Format**

An object of class `list` of length 6.

---

tidy_ranges	<i>Tidy the metadata of a GRanges object</i>
-------------	--

---

**Description**

Tidy the metadata of a GRanges object

**Usage**

```
tidy_ranges(gr, columns, low = 0)
```

**Arguments**

gr	A GRanges object
columns	A character vectors. The names of columns to be tidied.
low	An integer. All values $\leq$ this integer will be removed.

**Value**

A GRanges object

**Examples**

```
tidy_ranges(segment(test_obj, cell = 'K562')[[1]])
```

---

transitions_file	<i>Make transitions file name</i>
------------------	-----------------------------------

---

**Description**

Make transitions file name

**Usage**

```
transitions_file(numstates)
```

**Arguments**

numstates	An integer
-----------	------------

**Value**

A string

**Examples**

```
transitions_file(3)
```

# Index

- \* **datasets**
  - test\_obj, 28
  - test\_objs, 28
- .Binarize, 3
- .LearnModel, 3
- accessors, 4
- annotate\_segments, 7
- annotatePeak, 7
- assay, 27
  
- bamCount, 20
- binarize\_bam, 8
- binarize\_bed, 9
- bins, 27
- bins (accessors), 4
- bins, segmentation-method (accessors), 4
  
- cells (accessors), 4
- cells, segmentation-method (accessors), 4
- class:segmentation (segmentation), 26
- compare\_models, 10
- count\_reads\_ranges, 10
- counts, 27
- counts (accessors), 4
- counts, segmentation-method (accessors), 4
  
- emission, 27
- emission (accessors), 4
- emission, segmentation-method (accessors), 4
- emissions\_file, 11
- enrichment\_files, 11
  
- get\_frequency, 12
- get\_width, 13
- GRanges, 27
  
- learn\_model, 13, 26, 27
- likelihood (accessors), 4
  
- likelihood, segmentation-method (accessors), 4
  
- markers (accessors), 4
- markers, segmentation-method (accessors), 4
- merge\_segments\_bins, 15
- methods, 16
- model, 26
- model (accessors), 4
- model, segmentation-method (accessors), 4
- model\_file, 16
  
- overlap, 27
- overlap (accessors), 4
- overlap, segmentation-method (accessors), 4
- overlap\_files, 17
  
- plot\_heatmap, 18
  
- range\_bins, 18
- range\_counts, 19
- read\_bam\_file, 20
- read\_bins\_file, 20
- read\_cellmark\_file, 21
- read\_chromsize\_file, 22
- read\_emissions\_file, 22
- read\_enrichment\_file, 23
- read\_model\_file, 24
- read\_overlap\_file, 24
- read\_segements\_file, 25
- read\_transitions\_file, 26
- rowRanges, 27
  
- segment, 27
- segment (accessors), 4
- segment, segmentation-method (accessors), 4
- segmentation, 14, 26
- segmentation-class (segmentation), 26

segments\_files, [27](#)  
show, segmentation-method (methods), [16](#)  
states (accessors), [4](#)  
states, segmentation-method (accessors),  
[4](#)  
SummarizedExperiment, [27](#)  
  
TES, [27](#)  
TES (accessors), [4](#)  
TES, segmentation-method (accessors), [4](#)  
test\_obj, [28](#)  
test\_objs, [28](#)  
tidy\_ranges, [29](#)  
transition, [27](#)  
transition (accessors), [4](#)  
transition, segmentation-method  
(accessors), [4](#)  
transitions\_file, [29](#)  
TSS, [27](#)  
TSS (accessors), [4](#)  
TSS, segmentation-method (accessors), [4](#)