

# Package ‘tidyCoverage’

May 26, 2026

**Title** Extract and aggregate genomic coverage over features of interest

**Version** 1.9.0

**Date** 2023-11-09

**Description** `tidyCoverage` framework enables tidy manipulation of collections of genomic tracks and features using `tidySummarizedExperiment` methods. It facilitates the extraction, aggregation and visualization of genomic coverage over individual or thousands of genomic loci, relying on `CoverageExperiment` and `AggregatedCoverage` classes. This accelerates the integration of genomic track data in genomic analysis workflows.

**License** MIT + file LICENSE

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

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

**biocViews** Software, Sequencing, Coverage,

**Encoding** UTF-8

**Roxygen** list(markdown = TRUE)

**RoxygenNote** 7.3.3

**Depends** R (>= 4.3.0), SummarizedExperiment

**Imports** S4Vectors, IRanges, GenomicRanges, GenomeInfoDb, BiocParallel, BiocIO, rtracklayer, methods, tidyr, tibble, ggplot2, ggrastr, dplyr, fansi, pillar, rlang, scales, cli, purrr, vctrs, stats

**Suggests** tidySummarizedExperiment, plyranges, TxDb.Mmusculus.UCSC.mm10.knownGene, AnnotationHub, GenomicFeatures, BiocStyle, hues, knitr, rmarkdown, sessioninfo, testthat (>= 3.0.0)

**Config/testthat/edition** 3

**VignetteBuilder** knitr

**LazyData** false

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

**git\_branch** devel

**git\_last\_commit** c3c722d

**git\_last\_commit\_date** 2026-04-28

**Repository** Bioconductor 3.24

**Date/Publication** 2026-05-25

**Author** Jacques Serizay [aut, cre]

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

## Contents

tidyCoverage-package	2
AggregatedCoverage	3
as_tibble-AggregatedCoverage	3
CoverageExperiment	4
data	7
expand-CoverageExperiment	7
ggplot-tidyCoverage	8
show-tidyCoverage	11
<b>Index</b>	<b>12</b>

---

tidyCoverage-package    *tidyCoverage: Tidyomics-based analysis of coverage tracks over genomic features*

---

## Description

The tidyCoverage package provides a bridge between Bioconductor SummarizedExperiment objects and the tidyomics project.

## Author(s)

Jacques Serizay

## References

Serizay J, Koszul R., Epigenomics coverage data extraction and aggregation in R with tidyCoverage. *Bioinformatics* 40, btae487 (2024). doi:10.1093/bioinformatics/btae487

## See Also

Useful links:

- <https://github.com/js2264/tidyCoverage>
- Report bugs at <https://github.com/js2264/tidyCoverage/issues>

---

AggregatedCoverage    *aggregate*

---

**Description**

Bin coverage contained in a CoverageExperiment into an AggregatedCoverage object.

**Usage**

```
## S4 method for signature 'CoverageExperiment'  
aggregate(x, bin = 1, ...)
```

**Arguments**

x	a CoverageExperiment object
bin	an integer to bin each assay by. The width of the AggregatedCoverage object should be a multiple of bin.
...	ignored

**Value**

an AggregatedCoverage object

**Examples**

```
data(ce)  
aggregate(ce, bin = 10)
```

---

as\_tibble-AggregatedCoverage  
                          *as\_tibble*

---

**Description**

Coerce an CoverageExperiment or AggregatedCoverage object into a tibble

**Usage**

```
## S3 method for class 'AggregatedCoverage'  
as_tibble(x, ...)
```

**Arguments**

x	A data frame, list, matrix, or other object that could reasonably be coerced to a tibble.
...	Unused, for extensibility.

**Value**

tibble

**Row names**

The default behavior is to silently remove row names.

New code should explicitly convert row names to a new column using the `rownames` argument.

For existing code that relies on the retention of row names, call `pkgconfig::set_config("tibble::rownames" = NA)` in your script or in your package's `.onLoad()` function.

**Life cycle**

Using `as_tibble()` for vectors is superseded as of version 3.0.0, prefer the more expressive `as_tibble_row()` and `as_tibble_col()` variants for new code.

**See Also**

`tibble()` constructs a tibble from individual columns. `enframe()` converts a named vector to a tibble with a column of names and column of values. Name repair is implemented using `vctrs::vec_as_names()`.

**Examples**

```
data(ac)
as_tibble(ac)
```

---

CoverageExperiment      *CoverageExperiment*

---

**Description**

`CoverageExperiment` objects store coverages for individual tracks over different sets of features.

The coverage assay contains a separate matrix for each combination of track x features. `CoverageExperiment` objects are instantiated using the `CoverageExperiment()` function, and can be coarsened using the `coarsen()` function.

**Usage**

```
CoverageExperiment(tracks, features, ...)
```

```
coarsen(x, window, ...)
```

```
## S4 method for signature 'BigWigFileList,GRangesList'
CoverageExperiment(
  tracks,
  features,
  width = NULL,
  center = FALSE,
  scale = FALSE,
  ignore.strand = TRUE,
  window = 1,
  BPPARAM = BiocParallel::bpparam()
)
```

```
## S4 method for signature 'BigWigFileList,GRanges'
CoverageExperiment(tracks, features, ...)
```

```

## S4 method for signature 'BigWigFileList,list'
CoverageExperiment(tracks, features, ...)

## S4 method for signature 'BigWigFile,GRangesList'
CoverageExperiment(tracks, features, ...)

## S4 method for signature 'BigWigFile,GRanges'
CoverageExperiment(tracks, features, ...)

## S4 method for signature 'BigWigFile,list'
CoverageExperiment(tracks, features, ...)

## S4 method for signature 'list,GRangesList'
CoverageExperiment(
  tracks,
  features,
  width = NULL,
  center = FALSE,
  scale = FALSE,
  ignore.strand = TRUE,
  window = 1,
  BPPARAM = BiocParallel::bpparam()
)

## S4 method for signature 'list,GRanges'
CoverageExperiment(tracks, features, ...)

## S4 method for signature 'list,list'
CoverageExperiment(tracks, features, ...)

## S4 method for signature 'RleList,GRangesList'
CoverageExperiment(tracks, features, ...)

## S4 method for signature 'RleList,GRanges'
CoverageExperiment(tracks, features, ...)

## S4 method for signature 'RleList,list'
CoverageExperiment(tracks, features, ...)

## S4 method for signature 'CoverageExperiment'
coarsen(x, window = 1, BPPARAM = BiocParallel::bpparam())

```

### Arguments

tracks	A genomic track imported as a RleList or a <i>named</i> list of genomic tracks.
features	A set of features imported as GRanges or a <i>named</i> GRangesList.
...	Passed to the relevant method
x	a CoverageExperiment object
window	an integer to coarsen coverage by.
width	Width to resize each set of genomic features

scale, center Logical, whether to scale and/or center tracks prior to summarization  
 ignore.strand Logical, whether to not take the features strand information  
 BPPARAM Passed to BiocParallel.

## Value

A CoverageExperiment object

## Examples

```
library(rtracklayer)
library(purrr)
library(plyranges)
TSSs_bed <- system.file("extdata", "TSSs.bed", package = "tidyCoverage")
features <- import(TSSs_bed) |> filter(strand == '+')

#####
## 1. Creating a `CoverageExperiment` object from a single BigWigFile
#####

RNA_fwd <- system.file("extdata", "RNA.fwd.bw", package = "tidyCoverage")
tracks <- BigWigFile(RNA_fwd)
CoverageExperiment(tracks, features, width = 5000)

#####
## 2. Creating a `CoverageExperiment` object from a BigWigFileList
#####

RNA_rev <- system.file("extdata", "RNA.rev.bw", package = "tidyCoverage")
tracks <- BigWigFileList(list(RNA_fwd = RNA_fwd, RNA_rev = RNA_rev))
CoverageExperiment(tracks, features, width = 5000)

#####
## 3. Creating a `CoverageExperiment` object from imported bigwig files
#####

tracks <- list(
  RNA_fwd = system.file("extdata", "RNA.fwd.bw", package = "tidyCoverage"),
  RNA_rev = system.file("extdata", "RNA.rev.bw", package = "tidyCoverage")
) |> map(import, as = 'Rle')
CoverageExperiment(tracks, features, width = 5000)

#####
## 4. Correct for strandness when recovering coverage
#####

TSSs_bed <- system.file("extdata", "TSSs.bed", package = "tidyCoverage")
features <- list(
  TSS_fwd = import(TSSs_bed) |> filter(strand == '+'),
  TSS_rev = import(TSSs_bed) |> filter(strand == '-')
)
tracks <- list(
  RNA_fwd = system.file("extdata", "RNA.fwd.bw", package = "tidyCoverage"),
  RNA_rev = system.file("extdata", "RNA.rev.bw", package = "tidyCoverage")
) |> map(import, as = 'Rle')
CoverageExperiment(tracks, features, width = 5000, ignore.strand = FALSE)
```

```
#####
## Aggregating a `CoverageExperiment` object
#####
data(ce)
coarsen(ce, window = 10)
```

---

data *Example CoverageExperiment and AggregatedCoverage objects*

---

## Description

Two example objects are provided in the tidyCoverage package:

- ce: a CoverageExperiment dataset containing stranded RNA-seq coverage (forward and reverse) over Scc1 peaks ( $\pm$  1kb).
- ac: an AggregatedCoverage object obtained with aggregate(ce).

## Usage

```
data(ce)
```

```
data(ac)
```

## Format

CoverageExperiment object containing 1 features set and 2 tracks.

AggregatedCoverage object containing 1 features set and 2 tracks.

## Details

Data was generated in yeast (S288c) and aligned to reference R64-1-1.

---

```
expand-CoverageExperiment
      Expand a CoverageExperiment object
```

---

## Description

A CoverageExperiment object can be coerced into a tibble using the tidySummarizedExperiment package, but this will not turn each coverage matrix into a "long" format. The expand function provided here allows one to coerce a CoverageExperiment object into a long data frame, and adds the ranges and seqnames to the resulting tibble.

## Usage

```
## S3 method for class 'CoverageExperiment'
expand(data, ..., .name_repair = NULL)
```

**Arguments**

<code>data</code>	A data frame.
<code>...</code>	<p>&lt;data-masking&gt; Specification of columns to expand or complete. Columns can be atomic vectors or lists.</p> <ul style="list-style-type: none"> <li>• To find all unique combinations of <code>x</code>, <code>y</code> and <code>z</code>, including those not present in the data, supply each variable as a separate argument: <code>expand(df, x, y, z)</code> or <code>complete(df, x, y, z)</code>.</li> <li>• To find only the combinations that occur in the data, use <code>nesting</code>: <code>expand(df, nesting(x, y, z))</code>.</li> <li>• You can combine the two forms. For example, <code>expand(df, nesting(school_id, student_id), date)</code> would produce a row for each present school-student combination for all possible dates.</li> </ul> <p>When used with factors, <code>expand()</code> and <code>complete()</code> use the full set of levels, not just those that appear in the data. If you want to use only the values seen in the data, use <code>forcats::fct_drop()</code>.</p> <p>When used with continuous variables, you may need to fill in values that do not appear in the data: to do so use expressions like <code>year = 2010:2020</code> or <code>year = full_seq(year, 1)</code>.</p>
<code>.name_repair</code>	One of "check_unique", "unique", "universal", "minimal", "unique_quiet", or "universal_quiet". See <code>vec_as_names()</code> for the meaning of these options.

**Value**

a tibble object

**Grouped data frames**

With grouped data frames created by `dplyr::group_by()`, `expand()` operates *within* each group. Because of this, you cannot expand on a grouping column.

**See Also**

`complete()` to expand list objects. `expand_grid()` to input vectors rather than a data frame.

**Examples**

```
data(ce)
ce
expand(ce)
```

---

ggplot-tidyCoverage    *Plotting functions*

---

**Description**

Plotting functions for tidyCoverage objects. Two geoms are provided:

- `geom_coverage()`: for plotting coverages over individual loci.
- `geom_aggrcoverage()`: for plotting aggregated coverages with confidence intervals.

See the Details section for more information on the aesthetics used by each geom.

**Usage**

```

geom_aggrcoverage(
  mapping = NULL,
  data = NULL,
  ...,
  unit = c("kb", "Mb", "b"),
  ci = TRUE,
  grid = FALSE,
  na.rm = FALSE,
  show.legend = NA,
  inherit.aes = TRUE
)

geom_coverage(
  mapping = NULL,
  data = NULL,
  ...,
  type = c("area", "line"),
  unit = c("kb", "Mb", "b"),
  grid = FALSE,
  alpha = 0.6,
  na.rm = FALSE,
  show.legend = NA,
  inherit.aes = TRUE,
  raster = TRUE
)

scale_y_coverage()

scale_x_genome(unit = c("kb", "Mb", "b"))

```

**Arguments**

mapping	Set of aesthetic mappings created by <code>aes()</code> . By default, no color/fill aesthetic is specified, but they can be assigned to a variable with <code>mapping = aes(...)</code> . Note that x and y are automatically filled.
data	Data frame passed to <code>geom_*</code> . Typically a <code>CoverageExperiment</code> object (expanded to a tibble) or a <code>AggregatedCoverage</code> object.
..., na.rm, show.legend, inherit.aes	Argument passed to ggplot internal functions
unit	Rounding of x axis (any of <code>c('b', 'kb', 'Mb')</code> ).
ci	Should the confidence interval be plotted by <code>geom_aggrcoverage()</code> ? (default: TRUE)
grid	Should the plot grid by displayed? (default: FALSE).
type	Choose between "line" and "area" style for <code>geom_coverage()</code> .
alpha	Transparency level for <code>geom_coverage()</code> (default: 0.6).
raster	Should the plot be rasterized for faster rendering? (default: TRUE)

## Details

These geoms are drawn using `geom_line/ribbon/area()` so they support the same aesthetics: `colour`, `linetype` and `linewidth`. Both geoms also support the `unit` argument to control the x axis units (b, kb, Mb).

In addition, they each support additional arguments:

- `geom_coverage` uses a `type` argument to switch between line plot and area plots;
- `geom_aggrcoverage` uses a `ci` argument to toggle the confidence interval display.

## Value

A ggplot object

## Examples

```
library(rtracklayer)
library(plyranges)
library(ggplot2)
library(purrr)
TSSs_bed <- system.file("extdata", "TSSs.bed", package = "tidyCoverage")
features <- list(
  TSS_fwd = import(TSSs_bed) |> filter(strand == '+'),
  TSS_rev = import(TSSs_bed) |> filter(strand == '-'),
  conv_sites = import(system.file("extdata", "conv_transcription_loci.bed", package = "tidyCoverage"))
)
tracks <- list(
  RNA_fwd = system.file("extdata", "RNA.fwd.bw", package = "tidyCoverage"),
  RNA_rev = system.file("extdata", "RNA.rev.bw", package = "tidyCoverage"),
  Scc1 = system.file("extdata", "Scc1.bw", package = "tidyCoverage")
) |> map(import, as = 'Rle')
ce <- CoverageExperiment(tracks, features, width = 5000, center = TRUE, scale = TRUE)
ac <- aggregate(ce)

#####
## 1. Plotting aggregated coverage
#####

ac |>
  as_tibble() |>
  ggplot() +
  geom_aggrcoverage(aes(col = track)) +
  facet_grid(track ~ features) +
  geom_vline(xintercept = 0, color = 'black', linetype = 'dashed', linewidth = 0.5)

#####
## 2. Plotting track coverages over individual loci
#####

ce2 <- CoverageExperiment(
  tracks,
  GRangesList(list(locus1 = "II:400001-455000", locus2 = "IV:720001-775000")),
  window = 50
)
expand(ce2) |>
  mutate(coverage = ifelse(track != 'Scc1', scales::oob_squish(coverage, c(0, 50)), coverage)) |>
  ggplot() +
```

```
geom_coverage(aes(fill = track)) +  
facet_grid(track~features, scales = 'free')
```

---

`show-tidyCoverage`      *show method for CoverageExperiment and AggregatedCoverage objects*

---

### **Description**

show method for CoverageExperiment and AggregatedCoverage objects

### **Arguments**

`object`            a CoverageExperiment or AggregatedCoverage object  
`setup`            a setup object returned from `pillar::tbl_format_setup()`.

### **Value**

Prints a message to the console describing the contents of the CoverageExperiment or AggregatedCoverage objects.

### **Examples**

```
data(ce)  
print(ce)  
data(ac)  
print(ac)
```

# Index

- \* **datasets**
  - data, 7
- \* **internal**
  - tidyCoverage-package, 2
  - .onLoad(), 4
  
- ac (data), 7
- aggregate, CoverageExperiment-method
  - (AggregatedCoverage), 3
- AggregatedCoverage, 3
- as\_tibble-AggregatedCoverage, 3
- as\_tibble.AggregatedCoverage
  - (as\_tibble-AggregatedCoverage), 3
  
- ce (data), 7
- coarsen (CoverageExperiment), 4
- coarsen, CoverageExperiment-method
  - (CoverageExperiment), 4
- complete(), 8
- CoverageExperiment, 4
- CoverageExperiment, BigWigFile, GRanges-method
  - (CoverageExperiment), 4
- CoverageExperiment, BigWigFile, GRangesList-method
  - (CoverageExperiment), 4
- CoverageExperiment, BigWigFile, list-method
  - (CoverageExperiment), 4
- CoverageExperiment, BigWigFileList, GRanges-method
  - (CoverageExperiment), 4
- CoverageExperiment, BigWigFileList, GRangesList-method
  - (CoverageExperiment), 4
- CoverageExperiment, BigWigFileList, list-method
  - (CoverageExperiment), 4
- CoverageExperiment, list, GRanges-method
  - (CoverageExperiment), 4
- CoverageExperiment, list, GRangesList-method
  - (CoverageExperiment), 4
- CoverageExperiment, list, list-method
  - (CoverageExperiment), 4
- CoverageExperiment, RleList, GRanges-method
  - (CoverageExperiment), 4
- CoverageExperiment, RleList, GRangesList-method
  - (CoverageExperiment), 4
- CoverageExperiment, RleList, list-method
  - (CoverageExperiment), 4
  
- data, 7
- dplyr::group\_by(), 8
  
- enframe(), 4
- expand(), 8
- expand-CoverageExperiment, 7
- expand.CoverageExperiment
  - (expand-CoverageExperiment), 7
- expand\_grid(), 8
  
- geom\_aggrcoverage
  - (ggplot-tidyCoverage), 8
- geom\_coverage (ggplot-tidyCoverage), 8
- ggplot-tidyCoverage, 8
  
- pillar::tbl\_format\_setup(), 11
  
- scale\_x\_genome (ggplot-tidyCoverage), 8
- scale\_y\_coverage (ggplot-tidyCoverage), 8
  
- show, AggregatedCoverage-method
  - (show-tidyCoverage), 11
- show, CoverageExperiment-method
  - (show-tidyCoverage), 11
- show-tidyCoverage, 11
  
- tibble(), 4
- tidyCoverage (tidyCoverage-package), 2
- tidyCoverage-package, 2
  
- vctrs::vec\_as\_names(), 4
- vec\_as\_names(), 8