

Package ‘wigggleplotr’

January 30, 2026

Title Make read coverage plots from BigWig files

Version 1.34.0

Author Kaur Alasoo [aut, cre]

Maintainer Kaur Alasoo <kaur.alasoo@gmail.com>

Description Tools to visualise read coverage from sequencing experiments together with genomic annotations (genes, transcripts, peaks). Introns of long transcripts can be rescaled to a fixed length for better visualisation of exonic read coverage.

Depends R (>= 3.6)

Imports dplyr, ggplot2 (>= 2.2.0), GenomicRanges, rtracklayer, cowplot, assertthat, purrr, S4Vectors, IRanges, GenomeInfoDb

License Apache License 2.0

LazyData true

RoxygenNote 7.2.0

Encoding UTF-8

Suggests knitr, rmarkdown, biomaRt, GenomicFeatures, testthat, ensemblDb, EnsDb.Hsapiens.v86, org.Hs.eg.db, TxDb.Hsapiens.UCSC.hg38.knownGene, AnnotationDbi, AnnotationFilter

VignetteBuilder knitr

biocViews ImmunoOncology, Coverage, RNASeq, ChIPSeq, Sequencing, Visualization, GeneExpression, Transcription, AlternativeSplicing

git_url <https://git.bioconductor.org/packages/wigggleplotr>

git_branch RELEASE_3_22

git_last_commit 67a899c

git_last_commit_date 2025-10-29

Repository Bioconductor 3.22

Date/Publication 2026-01-29

Contents

extractCoverageData	2
getGenotypePalette	4
makeManhattanPlot	4
ncoa7_cdss	5
ncoa7_exons	5
ncoa7_metadata	6
pasteFactors	6
plotCoverage	7
plotCoverageData	9
plotCoverageFromEnsemblDb	10
plotCoverageFromUCSC	11
plotTranscripts	12
plotTranscriptsFromEnsemblDb	13
plotTranscriptsFromUCSC	14
wiggleplotr	15
Index	16

extractCoverageData	<i>Extract read coverage data from the bigWig files</i>
---------------------	---

Description

Does not work on Windows, because rtracklayer cannot read BigWig files on Windows.

Usage

```
extractCoverageData(
  exons,
  cdss = NULL,
  transcript_annotations = NULL,
  track_data,
  rescale_introns = TRUE,
  new_intron_length = 50,
  flanking_length = c(50, 50),
  plot_fraction = 0.1,
  mean_only = TRUE,
  region_coords = NULL
)
```

Arguments

exons	list of GRanges objects, each object containing exons for one transcript. The list must have names that correspond to transcript_id column in transcript_annotations data.frame.
cdss	list of GRanges objects, each object containing the coding regions (CDS) of a single transcript. The list must have names that correspond to transcript_id column in transcript_annotations data.frame. If cdss is not specified then exons list will be used for both arguments. (default: NULL).

transcript_annotations	Data frame with at least three columns: transcript_id, gene_name, strand. Used to construct transcript labels. (default: NULL)
track_data	data.frame with the metadata for the bigWig read coverage files. Must contain the following columns: • sample_id - unique id for each sample. • track_id - if multiple samples (bigWig files) have the same track_id they will be overlayed on the same plot, track_id is also used as the facet label on the right. • bigWig - path to the bigWig file. • scaling_factor - normalisation factor for each sample, useful if different samples sequenced to different depth and bigWig files not normalised for that. • colour_group - additional column to group samples into, is used as the colour of the coverage track.
rescale_introns	Specifies if the introns should be scaled to fixed length or not. (default: TRUE)
new_intron_length	length (bp) of introns after scaling. (default: 50)
flanking_length	Lengths of the flanking regions upstream and downstream of the gene. (default: c(50,50))
plot_fraction	Size of the random sub-sample of points used to plot coverage (between 0 and 1). Smaller values make plotting significantly faster. (default: 0.1)
mean_only	Plot only mean coverage within each combination of track_id and colour_group values. Useful for example for plotting mean coverage stratified by genotype (which is specified in the colour_group column) (default: TRUE).
region_coords	Start and end coordinates of the region to plot, overrides flanking_length parameter. The 'both' option tends to give better results for wide regions. (default: area).

Value

List containing all of the necessary data for the plotCoverageData function ()

Examples

```
require("dplyr")
require("GenomicRanges")
sample_data = dplyr::data_frame(sample_id = c("aipt_A", "aipt_C", "bima_A", "bima_C"),
  condition = factor(c("Naive", "LPS", "Naive", "LPS"), levels = c("Naive", "LPS")),
  scaling_factor = 1) %>%
  dplyr::mutate(bigWig = system.file("extdata", paste0(sample_id, ".str2.bw"), package = "wiggleplotr"))

track_data = dplyr::mutate(sample_data, track_id = condition, colour_group = condition)

selected_transcripts = c("ENST00000438495", "ENST00000392477") #Plot only two transcripts of the gens
## Not run:
extractCoverageData(ncoa7_exons[selected_transcripts], ncoa7_cdss[selected_transcripts], ncoa7_metadata, track_data)

## End(Not run)
```

getGenotypePalette	<i>Returns a three-colour palette suitable for visualising read coverage stratified by genotype</i>
--------------------	---

Description

Returns a three-colour palette suitable for visualising read coverage stratified by genotype

Usage

```
getGenotypePalette(old = FALSE)
```

Arguments

old	Return old colour palette (now deprecated).
-----	---

Value

Vector of three colours.

Examples

```
getGenotypePalette()
```

makeManhattanPlot	<i>Make a Manhattan plot of p-values</i>
-------------------	--

Description

The Manhattan plots is compatible with wiggpleplotr read coverage and transcript structure plots. Can be appended to those using the cowplot::plot_grid() function.

Usage

```
makeManhattanPlot(
  pvalues_df,
  region_coords,
  color_R2 = FALSE,
  data_track = TRUE
)
```

Arguments

pvalues_df	Data frame of association p-values (required columns: track_id, p_nominal, pos)
region_coords	Start and end coordinates of the region to plot.
color_R2	Color the points according to R2 from the lead variant. Require R2 column in the pvalues_df data frame.
data_track	If TRUE, then remove all information from x-axis. Makes it easy to append to read coverage or transcript structure plots using cowplot::plot_grid().

Value

ggplot2 object

Examples

```
data = dplyr::data_frame(track_id = "GWAS", pos = sample(c(1:1000), 200), p_nominal = runif(200, min = 0.000000  
makeManhattanPlot(data, c(1,1000), data_track = FALSE)
```

ncoa7_cdss

Coding sequences from 9 protein coding transcripts of NCOA7

Description

A dataset containing start and end coordinates of coding sequences (CDS) from nine protein coding transcripts of NCOA7.

Usage

ncoa7_cdss

Format

A GRangesList object with 9 elements:

element CDS start and end coordinates for a single transcript (GRanges object) ...

Source

<http://www.ensembl.org/>

ncoa7_exons

Exons from 9 protein coding transcripts of NCOA7

Description

A dataset containing start and end coordinates of exons from nine protein coding transcripts of NCOA7.

Usage

ncoa7_exons

Format

A GRangesList object with 9 elements:

element Exon start and end coordinates for a single transcript (GRanges object) ...

Source

<http://www.ensembl.org/>

ncoa7_metadata	<i>Gene metadata for NCOA7</i>
----------------	--------------------------------

Description

A a list of transcripts for NCOA7.

Usage

```
ncoa7_metadata
```

Format

A data.frame object with 4 columns:

transcript_id Ensembl transcript id.

gene_id Ensembl gene id.

gene_name Human readable gene name.

strand Strand of the transcript (either +1 or -1). ...

Source

<http://www.ensembl.org/>

pasteFactors	<i>Paste two factors together and preserved their joint order.</i>
--------------	--

Description

Paste two factors together and preserved their joint order.

Usage

```
pasteFactors(factor1, factor2)
```

Arguments

factor1 First factor

factor2 Second factor

Value

Factors factor1 and factor2 pasted together.

plotCoverage	<i>Plot read coverage across a genomic region</i>
--------------	---

Description

Also supports rescaling introns to constant length. Extracts read coverage from bigWig files with `extractCoverageData` and plots it with `plotCoverageData`. Custom visualisations can be created by modifying the `plotCoverageData` function. Does not work on Windows, because `rtracklayer` cannot read BigWig files on Windows.

Usage

```
plotCoverage(
  exons,
  cdss = NULL,
  transcript_annotatons = NULL,
  track_data,
  rescale_introns = TRUE,
  new_intron_length = 50,
  flanking_length = c(50, 50),
  plot_fraction = 0.1,
  heights = c(0.75, 0.25),
  alpha = 1,
  fill_palette = c("#a1dab4", "#41b6c4", "#225ea8"),
  mean_only = TRUE,
  connect_exons = TRUE,
  transcript_label = TRUE,
  return_subplots_list = FALSE,
  region_coords = NULL,
  coverage_type = "area",
  show_legend = FALSE
)
```

Arguments

- | | |
|------------------------------------|---|
| <code>exons</code> | list of GRanges objects, each object containing exons for one transcript. The list must have names that correspond to <code>transcript_id</code> column in <code>transcript_annotatons</code> data.frame. |
| <code>cdss</code> | list of GRanges objects, each object containing the coding regions (CDS) of a single transcript. The list must have names that correspond to <code>transcript_id</code> column in <code>transcript_annotatons</code> data.frame. If <code>cdss</code> is not specified then <code>exons</code> list will be used for both arguments. (default: NULL). |
| <code>transcript_annotatons</code> | Data frame with at least three columns: <code>transcript_id</code> , <code>gene_name</code> , <code>strand</code> . Used to construct transcript labels. (default: NULL) |
| <code>track_data</code> | data.frame with the metadata for the bigWig read coverage files. Must contain the following columns: <ul style="list-style-type: none"> <code>sample_id</code> - unique id for each sample. |

- track_id - if multiple samples (bigWig files) have the same track_id they will be overlayed on the same plot, track_id is also used as the facet label on the right.
- bigWig - path to the bigWig file.
- scaling_factor - normalisation factor for each sample, useful if different samples sequenced to different depth and bigWig files not normalised for that.
- colour_group - additional column to group samples into, is used as the colour of the coverage track.

rescale_introns	Specifies if the introns should be scaled to fixed length or not. (default: TRUE)
new_intron_length	length (bp) of introns after scaling. (default: 50)
flanking_length	Lengths of the flanking regions upstream and downstream of the gene. (default: c(50,50))
plot_fraction	Size of the random sub-sample of points used to plot coverage (between 0 and 1). Smaller values make plotting significantly faster. (default: 0.1)
heights	Specifies the proportion of the height that is dedicated to coverage plots (first value) relative to transcript annotations (second value). (default: c(0.75,0.25))
alpha	Transparency (alpha) value for the read coverage tracks. Useful to set to something < 1 when overlaying multiple tracks (see track_id). (default: 1)
fill_palette	Vector of fill colours used for the coverage tracks. Length must be equal to the number of unique values in track_data\$colour_group column.
mean_only	Plot only mean coverage within each combination of track_id and colour_group values. Useful for example for plotting mean coverage stratified by genotype (which is specified in the colour_group column) (default: TRUE).
connect_exons	Print lines that connect exons together. Set to FALSE when plotting peaks (default: TRUE).
transcript_label	If TRUE then transcript labels are printed above each transcript. (default: TRUE).
return_subplots_list	Instead of a joint plot return a list of subplots that can be joined together manually.
region_coords	Start and end coordinates of the region to plot, overrides flanking_length parameter.
coverage_type	Specifies if the read coverage is represented by either 'line', 'area' or 'both'. The 'both' option tends to give better results for wide regions. (default: area).
show_legend	display legend for the colour_group next to the read coverage plot (default: FALSE).

Value

Either object from cow_plot::plot_grid() function or a list of subplots (if return_subplots_list == TRUE)

Examples

```

require("dplyr")
require("GenomicRanges")
sample_data = dplyr::data_frame(sample_id = c("aibt_A", "aibt_C", "bima_A", "bima_C"),
  condition = factor(c("Naive", "LPS", "Naive", "LPS"), levels = c("Naive", "LPS")),
  scaling_factor = 1) %>%
  dplyr::mutate(bigWig = system.file("extdata", paste0(sample_id, ".str2.bw"), package = "wigglyplotr"))

track_data = dplyr::mutate(sample_data, track_id = condition, colour_group = condition)

selected_transcripts = c("ENST00000438495", "ENST00000392477") #Plot only two transcripts of the gens
## Not run:
plotCoverage(ncoa7_exons[selected_transcripts], ncoa7_cdss[selected_transcripts],
  ncoa7_metadata, track_data,
  heights = c(2,1), fill_palette = getGenotypePalette())

## End(Not run)

```

plotCoverageData	<i>Plot read coverage across a genomic region</i>
------------------	---

Description

Does not work on Windows, because rtracklayer cannot read BigWig files on Windows.

Usage

```

plotCoverageData(
  coverage_data_list,
  heights = c(0.75, 0.25),
  alpha = 1,
  fill_palette = c("#a1dab4", "#41b6c4", "#225ea8"),
  connect_exons = TRUE,
  transcript_label = TRUE,
  return_subplots_list = FALSE,
  coverage_type = "area",
  show_legend = FALSE
)

```

Arguments

coverage_data_list

List of required from the extractCoverageData function:

- exons - list of GRanges objects, each object containing exons for one transcript.
- cdss - list of GRanges objects, each object containing the coding regions (CDS) of a single transcript.
- plotting_annotations - Transcript labels for plotting.
- tx_annotations - Transcript coordinates for plotting.
- xlabel - Label of the x-axis.

	• coverage_df - Read coverage data frame. • limits - x-axis limits.
heights	Specifies the proportion of the height that is dedicated to coverage plots (first value) relative to transcript annotations (second value). (default: c(0.75,0.25))
alpha	Transparency (alpha) value for the read coverage tracks. Useful to set to something < 1 when overlaying multiple tracks (see track_id). (default: 1)
fill_palette	Vector of fill colours used for the coverage tracks. Length must be equal to the number of unique values in track_data\$colour_group column.
connect_exons	Print lines that connect exons together. Set to FALSE when plotting peaks (default: TRUE).
transcript_label	If TRUE then transcript labels are printed above each transcript. (default: TRUE).
return_subplots_list	Instead of a joint plot return a list of subplots that can be joined together manually.
coverage_type	Specifies if the read coverage is represented by either 'line', 'area' or 'both'. The 'both' option tends to give better results for wide regions. (default: area).
show_legend	display legend for the colour_group next to the read coverage plot (default: FALSE).

Value

Either object from cow_plot::plot_grid() function or a list of subplots (if return_subplots_list == TRUE)

Examples

```
require("dplyr")
require("GenomicRanges")
sample_data = dplyr::data_frame(sample_id = c("aibt_A", "aibt_C", "bima_A", "bima_C"),
  condition = factor(c("Naive", "LPS", "Naive", "LPS"), levels = c("Naive", "LPS")),
  scaling_factor = 1) %>%
  dplyr::mutate(bigWig = system.file("extdata", paste0(sample_id, ".str2.bw"), package = "wigglyplotr"))

track_data = dplyr::mutate(sample_data, track_id = condition, colour_group = condition)

selected_transcripts = c("ENST00000438495", "ENST00000392477") #Plot only two transcripts of the gens
## Not run:
cov_data = extractCoverageData(ncoa7_exons[selected_transcripts], ncoa7_cdss[selected_transcripts], ncoa7_me
plotCoverageData(cov_data, heights = c(2,1), fill_palette = getGenotypePalette())

## End(Not run)
```

plotCoverageFromEnsemblDb

Plot read coverage directly from ensemblDb object.

Description

A wrapper around the plotCoverage function. See the documentation for ([plotCoverage](#)) for more information.

Usage

```
plotCoverageFromEnsemblDb(ensemldb, gene_names, transcript_ids = NULL, ...)
```

Arguments

ensemldb	ensemldb object.
gene_names	List of gene names to be plotted.
transcript_ids	Optional list of transcript ids to be plotted.
...	Additional parameters to be passed to plotCoverage.

Value

ggplot2 object

Examples

```
require("EnsDb.Hsapiens.v86")
require("dplyr")
require("GenomicRanges")
sample_data = dplyr::data_frame(sample_id = c("aibt_A", "aibt_C", "bima_A", "bima_C"),
  condition = factor(c("Naive", "LPS", "Naive", "LPS"), levels = c("Naive", "LPS")),
  scaling_factor = 1) %>%
dplyr::mutate(bigWig = system.file("extdata", paste0(sample_id, ".str2.bw"), package = "wigglyplotr"))

track_data = dplyr::mutate(sample_data, track_id = condition, colour_group = condition)
## Not run:
plotCoverageFromEnsemblDb(EnsDb.Hsapiens.v86, "NCOA7", transcript_ids = c("ENST00000438495", "ENST0000039247"),
  track_data, heights = c(2,1), fill_palette = getGenotypePalette())
## End(Not run)
```

plotCoverageFromUCSC *Plot read coverage directly from UCSC OrgDb and TxDb objects.*

Description

A wrapper around the plotCoverage function. See the documentation for ([plotCoverage](#)) for more information.

Usage

```
plotCoverageFromUCSC(orgdb, txdb, gene_names, transcript_ids = NULL, ...)
```

Arguments

orgdb UCSC OrgDb object.
txdb UCSC TxDb object.
gene_names List of gene names to be plotted.
transcript_ids Optional list of transcript ids to be plotted.
... Additional parameters to be passed to plotCoverage.

Value

ggplot2 object

Examples

```
require("dplyr")
require("GenomicRanges")
require("org.Hs.eg.db")
require("TxDb.Hsapiens.UCSC.hg38.knownGene")

orgdb = org.Hs.eg.db
txdb = TxDb.Hsapiens.UCSC.hg38.knownGene

sample_data = dplyr::data_frame(sample_id = c("aibt_A", "aibt_C", "bima_A", "bima_C"),
  condition = factor(c("Naive", "LPS", "Naive", "LPS"), levels = c("Naive", "LPS")),
  scaling_factor = 1) %>%
dplyr::mutate(bigWig = system.file("extdata", paste0(sample_id, ".str2.bw"), package = "wigglyplotr"))

track_data = dplyr::mutate(sample_data, track_id = condition, colour_group = condition)
## Not run:
#Note: This example does not work, because UCSC and Ensembl use different chromosome names
plotCoverageFromUCSC(orgdb, txdb, "NCOA7", transcript_ids = c("ENST00000438495.6", "ENST00000368357.7"),
track_data, heights = c(2,1), fill_palette = getGenotypePalette())

## End(Not run)
```

plotTranscripts

Quickly plot transcript structure without read coverage tracks

Description

Quickly plot transcript structure without read coverage tracks

Usage

```
plotTranscripts(
  exons,
  cdss = NULL,
  transcript_annotations = NULL,
  rescale_introns = TRUE,
  new_intron_length = 50,
  flanking_length = c(50, 50),
  connect_exons = TRUE,
```

```

    transcript_label = TRUE,
    region_coords = NULL
)

```

Arguments

exons	list of GRanges objects, each object containing exons for one transcript. The list must have names that correspond to transcript_id column in transcript_annotations data.frame.
cdss	list of GRanges objects, each object containing the coding regions (CDS) of a single transcript. The list must have names that correspond to transcript_id column in transcript_annotations data.frame. If cdss is not specified then exons list will be used for both arguments. (default: NULL)
transcript_annotations	Data frame with at least three columns: transcript_id, gene_name, strand. Used to construct transcript labels. (default: NULL)
rescale_introns	Specifies if the introns should be scaled to fixed length or not. (default: TRUE)
new_intron_length	length (bp) of introns after scaling. (default: 50)
flanking_length	Lengths of the flanking regions upstream and downstream of the gene. (default: c(50,50))
connect_exons	Print lines that connect exons together. Set to FALSE when plotting peaks (default: TRUE).
transcript_label	If TRUE then transcript labels are printed above each transcript. (default: TRUE).
region_coords	Start and end coordinates of the region to plot, overrides flanking_length parameter.

Value

ggplot2 object

Examples

```
plotTranscripts(ncoa7_exons, ncoa7_cdss, ncoa7_metadata, rescale_introns = FALSE)
```

```
plotTranscriptsFromEnsemblDb
```

Plot transcripts directly from ensemblDb object.

Description

A wrapper around the plotTranscripts function. See the documentation for ([plotTranscripts](#)) for more information.

Usage

```
plotTranscriptsFromEnsemblDb(ensemldb, gene_names, transcript_ids = NULL, ...)
```

Arguments

ensemldb	ensemldb object.
gene_names	List of gene names to be plotted.
transcript_ids	Optional list of transcript ids to be plotted.
...	Additional parameters to be passed to plotTranscripts

Value

ggplot2 object

Examples

```
require("EnsDb.Hsapiens.v86")
plotTranscriptsFromEnsemblDb(EnsDb.Hsapiens.v86, "NCOA7", transcript_ids = c("ENST00000438495", "ENST0000039
```

```
plotTranscriptsFromUCSC
```

Plot transcripts directly from UCSC OrgDb and TxDb objects.

Description

A wrapper around the plotTranscripts function. See the documentation for ([plotTranscripts](#)) for more information. Note that this function is much slower than ([plotTranscripts](#)) or ([plotTranscriptsFromEnsemblDb](#)) functions, because individually extracting exon coordinates from txdb objects is quite inefficient.

Usage

```
plotTranscriptsFromUCSC(orgdb, txdb, gene_names, transcript_ids = NULL, ...)
```

Arguments

orgdb	UCSC OrgDb object.
txdb	UCSC TxDb object.
gene_names	List of gene names to be plot.
transcript_ids	Optional list of transcript ids to be plot. (default = NULL)
...	Additional parameters to be passed to plotTranscripts

Value

Transcript plot.

Examples

```
#Load OrgDb and TxDb objects with UCSC gene annotations
require("org.Hs.eg.db")
require("TxDb.Hsapiens.UCSC.hg38.knownGene")
orgdb = org.Hs.eg.db
txdb = TxDb.Hsapiens.UCSC.hg38.knownGene

plotTranscriptsFromUCSC(orgdb, txdb, "NCOA7", transcript_ids = c("ENST00000438495.6", "ENST00000368357.7"))
```

`wigglyplotr`*wigglyplotr*

Description

wigglyplotr package provides tools to visualise transcript annotations ([plotTranscripts](#)) and plot sequencing read coverage over annotated transcripts ([plotCoverage](#)).

Details

You can also use convenient wrapper functions ([plotTranscriptsFromEnsembl](#)), ([plotCoverageFromEnsembl](#)), ([plotTranscriptsFromUCSC](#)) and ([plotCoverageFromUCSC](#)).

To learn more about wigglyplotr, start with the vignette: `browseVignettes(package = "wigglyplotr")`

Index

* datasets

ncoa7_cdss, [5](#)

ncoa7_exons, [5](#)

ncoa7_metadata, [6](#)

extractCoverageData, [2](#)

getGenotypePalette, [4](#)

makeManhattanPlot, [4](#)

ncoa7_cdss, [5](#)

ncoa7_exons, [5](#)

ncoa7_metadata, [6](#)

pasteFactors, [6](#)

plotCoverage, [7](#), [11](#), [15](#)

plotCoverageData, [9](#)

plotCoverageFromEnsemblDb, [10](#), [15](#)

plotCoverageFromUCSC, [11](#), [15](#)

plotTranscripts, [12](#), [13–15](#)

plotTranscriptsFromEnsemblDb, [13](#), [14](#), [15](#)

plotTranscriptsFromUCSC, [14](#), [15](#)

wiggleplotr, [15](#)