

Package ‘hidecan’

May 8, 2026

Title Create HIDEKAN Plots for Visualising Genome-Wide Association Studies and Differential Expression Results

Version 1.1.0

Description Generates HIDEKAN plots that summarise and combine the results of genome-wide association studies (GWAS) and transcriptomics differential expression analyses (DE), along with manually curated candidate genes of interest. The HIDEKAN plot is presented in Angelin-Bonnet et al. (2023) (currently in review).

License MIT + file LICENSE

URL <https://plantandfoodresearch.github.io/hidecan/>,
<https://github.com/PlantandFoodResearch/hidecan>

BugReports <https://github.com/PlantandFoodResearch/hidecan/issues>

Encoding UTF-8

RoxygenNote 7.2.3

Depends R (>= 3.5.0)

Imports dplyr, ggnewscale, ggplot2, ggrepel, purrr, shiny, tibble,
tidyr, viridis, vroom

Suggests knitr, rmarkdown, stringr, testthat (>= 3.0.0)

Config/testthat/edition 3

VignetteBuilder knitr

Language en-GB

NeedsCompilation no

Author Olivia Angelin-Bonnet [aut, cre] (ORCID:
<<https://orcid.org/0000-0002-7708-2919>>)

Maintainer Olivia Angelin-Bonnet <olivia.angelin-bonnet@plantandfood.co.nz>

Repository CRAN

Date/Publication 2023-02-10 09:40:02 UTC

Contents

.check_cols	2
.compute_chrom_length_genes	3
apply_threshold	3
CAN_data	4
combine_chrom_length	5
compute_chrom_length	6
create_hidecan_plot	6
DE_data	9
get_example_data	10
GWAS_data	10
GWAS_data_from_gwaspoly	11
hidecan_plot	12
hidecan_plot_from_gwaspoly	15
new_CAN_data	15
new_DE_data	16
new_GWAS_data	16
run_hidecan_shiny	17
validate_CAN_data	17
validate_DE_data	18
validate_GWAS_data	18

Index	19
--------------	-----------

.check_cols	<i>Checks whether some columns are present in a tibble</i>
-------------	--

Description

Checks whether some columns are present in a tibble

Usage

```
.check_cols(x, col_names, param_name = "Input data-frame")
```

Arguments

x	Tibble
col_names	character vector of column names
param_name	Character, name of the dataframe to use in the error message.

Value

invisible NULL

```
.compute_chrom_length_genes
```

Computes chromosomes' length for a tibble of genes

Description

Computes the length (in bp) of each chromosome as the maximum position of genes on the chromosome.

Usage

```
.compute_chrom_length_genes(x)
```

Arguments

x Either a DE_data or CAN_data object.

Value

A tibble with two columns: chromosome (chromosome name) and length (chromosome length in base pair).

```
apply_threshold            Filters GWAS or DE results based on a threshold
```

Description

Filters markers or genes/transcripts based on a threshold applied to their GWAS or DE score, and log2(fold-change) (if applicable). For a set of candidate genes, simply returns the list. Note that markers or genes with a missing score or log2(fold-change) will be removed from the dataset.

Usage

```
apply_threshold(x, score_thr = 0, log2fc_thr = 0)
```

```
## S3 method for class 'GWAS_data'  
apply_threshold(x, score_thr = 0, log2fc_thr = 0)
```

```
## S3 method for class 'DE_data'  
apply_threshold(x, score_thr = 0, log2fc_thr = 0)
```

```
## S3 method for class 'CAN_data'  
apply_threshold(x, score_thr = 0, log2fc_thr = 0)
```

```
## Default S3 method:  
apply_threshold(x, score_thr = 0, log2fc_thr = 0)
```

Arguments

x	Either a GWAS_data, DE_data or CAN_data object.
score_thr	Numeric, threshold to use on markers' or genes/transcripts' score. Only markers or genes with a score equal to or higher than this threshold will be retained. Default value is 0. Ignored for CAN_data.
log2fc_thr	Numeric, threshold to use on the absolute value of genes/ transcripts' log2(fold-change). Only genes/transcripts with an absolute log2(fold-change) equal to or higher than this threshold will be retained. Ignored for GWAS_data and CAN_data.

Value

A filtered tibble (of class GWAS_data_thr, DE_data_thr or CAN_data_thr).

Examples

```
x <- get_example_data()

## For GWAS results
apply_threshold(GWAS_data(x[["GWAS"]]), score_thr = 4)

## For DE results - in second line, no threshold is applied
## on the log2(fold-change)
apply_threshold(DE_data(x[["DE"]]), score_thr = -log10(0.05), log2fc_thr = 1)
apply_threshold(DE_data(x[["DE"]]), score_thr = -log10(0.05), log2fc_thr = 0)

## No effect on the Candidate genes
apply_threshold(CAN_data(x[["CAN"]]))
```

CAN_data	<i>Creates a CAN_data object</i>
----------	----------------------------------

Description

Creates a CAN_data object from a tibble or data-frame of candidate genes.

Usage

```
CAN_data(dat, keep_rownames_as = NULL)
```

Arguments

dat	Tibble, set of candidate genes of interest. See Details.
keep_rownames_as	Character, the name of the column in which to save the rownames of the input data-frame. Default value is NULL, i.e. rownames will be discarded.

Details

The input data should have one row per gene, and at least the following columns:

- chromosome: character column, chromosome on which the gene is located.
- start and end: numeric, starting and end position of the gene (in bp). A column position will be constructed as the middle value (mean) between start and end.
- name: character, the name of the candidate genes to be displayed.

Value

A CAN_data object, i.e. a tibble.

Examples

```
x <- get_example_data()
CAN_data(x[["CAN"]])
```

combine_chrom_length *Computes chromosomes' length from list*

Description

Computes the length (in bp) of each chromosome from a list of GWAS and DE results as well as candidate gene lists.

Usage

```
combine_chrom_length(x)
```

Arguments

x A list of GWAS_data, DE_data or CAN_data objects.

Value

A tibble with two columns: chromosome (chromosome name) and length (chromosome length in base pair).

Examples

```
x <- get_example_data()
y <- list("GWAS" = GWAS_data(x[["GWAS"]]),
         "DE" = DE_data(x[["DE"]]),
         "CAN" = CAN_data(x[["CAN"]]))

combine_chrom_length(y)
```

compute_chrom_length *Computes chromosomes' length*

Description

Computes the length (in bp) of each chromosome as the maximum position of markers or genes on the chromosome.

Usage

```
compute_chrom_length(x)

## S3 method for class 'GWAS_data'
compute_chrom_length(x)

## S3 method for class 'DE_data'
compute_chrom_length(x)

## S3 method for class 'CAN_data'
compute_chrom_length(x)
```

Arguments

x Either a GWAS_data, DE_data or CAN_data object.

Value

A tibble with two columns: chromosome (chromosome name) and length (chromosome length in base pair).

Examples

```
x <- get_example_data()

compute_chrom_length(GWAS_data(x[["GWAS"]]))
compute_chrom_length(DE_data(x[["DE"]]))
compute_chrom_length(CAN_data(x[["CAN"]]))
```

create_hidecan_plot *Creates a HIDECAN plot*

Description

Creates a HIDECAN plot from a list of filtered GWAS or DE results and/or candidate genes.

Usage

```

create_hidecan_plot(
  x,
  chrom_length,
  colour_genes_by_score = TRUE,
  remove_empty_chrom = FALSE,
  chroms = NULL,
  chrom_limits = NULL,
  title = NULL,
  subtitle = NULL,
  n_rows = NULL,
  n_cols = 2,
  legend_position = "bottom",
  point_size = 3,
  label_size = 3.5,
  label_padding = 0.15
)

```

Arguments

x	A list of GWAS_data_thr, DE_data_thr and/or CAN_data_thr produced by the apply_threshold() function. If named, the names will be appended to the y-axis labels (use ' ' as empty name in the list).
chrom_length	Tibble with columns chromosome and length, giving for each chromosome its length in bp (see combine_chrom_length() function).
colour_genes_by_score	Logical, whether to colour the genes by score (TRUE) or by log2(fold-change) (FALSE). Default value is TRUE.
remove_empty_chrom	Logical, should chromosomes with no significant markers/genes nor candidate genes be removed from the plot? Default value if FALSE.
chroms	Character vector, name of chromosomes to include in the plot.
chrom_limits	Integer vector of length 2, or named list where the elements are integer vectors of length 2. If vector, gives the lower and upper limit of the chromosomes (in bp) to use in the plot. If a named list, names should correspond to chromosome names. Gives for each chromosome the lower and upper limits (in bp) to use in the plot. Doesn't have to be specified for all chromosomes. Default value is NULL, i.e. no limits are applied to the chromosomes (they will be plotted in their entirety).
title	Character, title of the plot. Default value is NULL (i.e. no title will be added to the plot).
subtitle	Character, subtitle of the plot. Default value is NULL (i.e. no subtitle will be added to the plot).
n_rows	Integer, number of rows of chromosomes to create in the plot. Default value is NULL.

n_cols	Integer, number of columns of chromosomes to create in the plot. Default value is 2. Will be set to NULL if n_rows is not NULL.
legend_position	Character, position of the legend in the plot. Can be bottom (default value), top, right, left or none.
point_size	Numeric, size of the points in the plot. Default value is 3.
label_size	Numeric, size of the gene labels in the plot. Default value is 3.5 (for <code>geom_label_repel</code>).
label_padding	Numeric, amount of padding around gene labels in the plot, as unit or number. Default value is 0.15 (for <code>geom_label_repel</code>).

Value

A ggplot.

Examples

```

if (interactive()) {
x <- get_example_data()
y <- list("GWAS" = GWAS_data(x[["GWAS"]]),
         "DE" = DE_data(x[["DE"]]),
         "CAN" = CAN_data(x[["CAN"]]))

chrom_length <- combine_chrom_length(y)

z <- list(
  apply_threshold(y[["GWAS"]], score_thr = 4),
  apply_threshold(y[["DE"]], score_thr = 1.3, log2fc_thr = 0.5),
  apply_threshold(y[["CAN"]])
)

create_hidecan_plot(z,
                   chrom_length,
                   label_size = 2)

## Colour genes according to their fold-change
create_hidecan_plot(z,
                   chrom_length,
                   colour_genes_by_score = FALSE,
                   label_size = 2)

## Add names to the datasets
create_hidecan_plot(setNames(z, c("Genomics", "RNAseq", "My list")),
                   chrom_length,
                   colour_genes_by_score = FALSE,
                   label_size = 2)

## Add names to some of the datasets only (e.g. not for GWAS results)
create_hidecan_plot(setNames(z, c(" ", "RNAseq", "My list")),
                   chrom_length,
                   colour_genes_by_score = FALSE,
                   label_size = 2)

```

```
## Set limits on all chromosomes (to "zoom in" to the 10-20Mb region)
create_hidecan_plot(z,
  chrom_length,
  label_size = 2,
  chrom_limits = c(10e6, 20e6))

## Set limits on some chromosomes only
create_hidecan_plot(z,
  chrom_length,
  label_size = 2,
  chrom_limits = list("ST4.03ch00" = c(10e6, 20e6),
    "ST4.03ch02" = c(15e6, 25e6)))
}
```

DE_data	<i>Creates a DE_data object</i>
---------	---------------------------------

Description

Creates a DE_data object from a tibble or data-frame of differential expression results.

Usage

```
DE_data(dat, keep_rownames_as = NULL)
```

Arguments

dat Tibble, results from a differential expression analysis. See Details.

keep_rownames_as Character, the name of the column in which to save the rownames of the input data-frame. Default value is NULL, i.e. rownames will be discarded.

Details

The input data should have one row per gene or transcript, and at least the following columns:

- **chromosome**: character column, chromosome on which the gene/transcript is located.
- **start and end**: numeric, starting and end position of the gene/transcript (in bp). A column position will be constructed as the middle value (mean) between start and end.
- **score or padj**: numeric, the DE score or adjusted p-value of the gene/transcript. If column score column is missing, will be constructed as $-\log_{10}(\text{padj})$.
- **foldChange or log2FoldChange**: numeric, the fold-change or $\log_2(\text{fold-change})$ of the gene/transcript. If column log2FoldChange is missing, will be constructed as $\log_2(\text{foldChange})$.

Value

A DE_data object, i.e. a tibble.

Examples

```
x <- get_example_data()
DE_data(x[["DE"]])
```

get_example_data	<i>Example dataset</i>
------------------	------------------------

Description

Returns a list of example datasets.

Usage

```
get_example_data()
```

Value

A list with the following elements:

- GWAS: a tibble of GWAS results, with columns id, chromosome, position and score.
- DE: a tibble of differential expression results, with columns gene, chromosome, padj, log2FoldChange, start, end and label.
- CAN: a tibble of candidate genes, with columns id, chromosome, start, end, name and gene_name.

GWAS_data	<i>Creates a GWAS_data object</i>
-----------	-----------------------------------

Description

Creates a GWAS_data object from a tibble or data-frame of GWAS results.

Usage

```
GWAS_data(dat, keep_rownames_as = NULL)
```

Arguments

dat	Tibble, results from a GWAS analysis. See Details.
keep_rownames_as	Character, the name of the column in which to save the rownames of the input data-frame. Default value is NULL, i.e. rownames will be discarded.

Details

The input data should have one row per marker, and at least the following columns:

- `chromosome`: character column, chromosome on which the marker is located.
- `position`: numeric, the physical position of the marker along the chromosome (in bp).
- `score` or `padj`: numeric, the GWAS score or adjusted p-value of the marker. If column `score` is missing, will be constructed as $-\log_{10}(\text{padj})$.

Value

A `GWAS_data` object, i.e. a tibble.

Examples

```
x <- get_example_data()
GWAS_data(x[["GWAS"]])
```

`GWAS_data_from_gwaspoly`

Extracts information from GWASpoly output

Description

Extracts GWAS results and chromosome length from `GWASpoly` output.

Usage

```
GWAS_data_from_gwaspoly(gwaspoly_output, traits = NULL, models = NULL)
```

Arguments

<code>gwaspoly_output</code>	A <code>GWASpoly.fitted</code> or <code>GWASpoly.thresh</code> object (returned by <code>GWASpoly::GWASpoly()</code> or <code>GWASpoly::set.threshold()</code> functions).
<code>traits</code>	Character vector, traits for which GWAS results should be extracted. If <code>NULL</code> (default value), all traits present are considered.
<code>models</code>	Character vector, genetic models for which GWAS results should be extracted. If <code>NULL</code> (default value), all genetic models present are considered.

Value

A list with the following elements:

- `gwas_data_list`: A named list of `GWAS_data` objects, giving the markers score for each possible trait/genetic model combination. The names of the list are in the form `trait (genetic model)`.
- `gwas_data_thr_list`: if the input data is a `GWASpoly.thresh` object (from the `GWASpoly::set.threshold()` function), a named list of `Gwas_data_thr`, with the significant markers score for each possible trait/genetic model combination. The names of the list are in the form `trait (genetic model)`.
- `chrom_length`: A tibble with one row per chromosome, giving the length (in bp) of each chromosome.

`hidecan_plot`
Wrapper to create a HIDECAN plot

Description

Wrapper function to create a HIDECAN plot from GWAS results, DE results or candidate genes.

Usage

```
hidecan_plot(
  gwas_list = NULL,
  de_list = NULL,
  can_list = NULL,
  score_thr_gwas = 4,
  score_thr_de = 2,
  log2fc_thr = 1,
  chrom_length = NULL,
  colour_genes_by_score = TRUE,
  remove_empty_chrom = FALSE,
  chroms = NULL,
  chrom_limits = NULL,
  title = NULL,
  subtitle = NULL,
  n_rows = NULL,
  n_cols = 2,
  legend_position = "bottom",
  point_size = 3,
  label_size = 3.5,
  label_padding = 0.15
)
```

Arguments

`gwas_list` Data-frame or list of data-frames containing GWAS results, each with at least a chromosome, position and either `padj` or `score` columns. If a named list, the names will be used in the plot.

de_list	Data-frame or list of data-frames containing DE results, each with at least a chromosome, start, end, log2FoldChange and either padj or score columns. If a named list, the names will be used in the plot.
can_list	Data-frame or list of data-frames containing candidate genes, each with at least a chromosome, start, end and name columns. If a named list, the names will be used in the plot.
score_thr_gwas	Numeric, the score threshold for GWAS results that will be used to select which markers will be plotted. Default value is 4.
score_thr_de	Numeric, the score threshold for DE results that will be used to select which markers will be plotted. Default value is 2.
log2fc_thr	Numeric, the log2(fold-change) threshold that will be used to select which genes will be plotted. Default value is 1.
chrom_length	Optional, tibble with columns chromosome and length, giving for each chromosome its length in bp. If NULL (the default), will be inferred from the GWAS, DE and candidate gene data.
colour_genes_by_score	Logical, whether to colour the genes by score (TRUE) or by log2(fold-change) (FALSE). Default value is TRUE.
remove_empty_chrom	Logical, should chromosomes with no significant markers/genes nor candidate genes be removed from the plot? Default value if FALSE.
chroms	Character vector, name of chromosomes to include in the plot.
chrom_limits	Integer vector of length 2, or named list where the elements are integer vectors of length 2. If vector, gives the lower and upper limit of the chromosomes (in bp) to use in the plot. If a named list, names should correspond to chromosome names. Gives for each chromosome the lower and upper limits (in bp) to use in the plot. Doesn't have to be specified for all chromosomes. Default value is NULL, i.e. no limits are applied to the chromosomes (they will be plotted in their entirety).
title	Character, title of the plot. Default value is NULL (i.e. no title will be added to the plot).
subtitle	Character, subtitle of the plot. Default value is NULL (i.e. no subtitle will be added to the plot).
n_rows	Integer, number of rows of chromosomes to create in the plot. Default value is NULL.
n_cols	Integer, number of columns of chromosomes to create in the plot. Default value is 2. Will be set to NULL if n_rows is not NULL.
legend_position	Character, position of the legend in the plot. Can be bottom (default value), top, right, left or none.
point_size	Numeric, size of the points in the plot. Default value is 3.
label_size	Numeric, size of the gene labels in the plot. Default value is 3.5 (for geom_label_repel).
label_padding	Numeric, amount of padding around gene labels in the plot, as unit or number. Default value is 0.15 (for geom_label_repel).

 hidecan_plot_from_gwaspoly

Creates a HIDECAN plot from GWASpoly output

Description

Creates a HIDECAN plot from the GWAS results from GWASpoly.

Usage

```
hidecan_plot_from_gwaspoly(gwaspoly_output, traits = NULL, models = NULL, ...)
```

Arguments

gwaspoly_output	A <code>GWASpoly.thresh</code> object (returned by the <code>GWASpoly::set.threshold()</code> function).
traits	Character vector, traits for which GWAS results should be extracted. If NULL (default value), all traits present are considered.
models	Character vector, genetic models for which GWAS results should be extracted. If NULL (default value), all genetic models present are considered.
...	Further arguments passed to the <code>create_hidecan_plot()</code> function.

Value

A `ggplot`.

 new_CAN_data

CAN_data constructor

Description

CAN_data constructor

Usage

```
new_CAN_data(dat)
```

Arguments

dat	Tibble, containing information about genes of interest, with at least columns chromosome, start, end, position and name.
-----	--

Value

A `CAN_data` object, i.e. a tibble.

new_DE_data	DE_data <i>constructor</i>
-------------	----------------------------

Description

DE_data constructor

Usage

new_DE_data(dat)

Arguments

dat	Tibble, results from a differential expression analysis, with at least columns chromosome, score, log2FoldChange, start, end and position.
-----	--

Value

A DE_data object, i.e. a tibble.

new_GWAS_data	GWAS_data <i>constructor</i>
---------------	------------------------------

Description

GWAS_data constructor

Usage

new_GWAS_data(dat)

Arguments

dat	Tibble, results from a GWAS analysis, with at least columns chromosome, position and score.
-----	---

Value

A GWAS_data object, i.e. a tibble.

run_hidecan_shiny	<i>Launches the HIDECAN shiny app</i>
-------------------	---------------------------------------

Description

Starts the HIDECAN shiny app. The app reads in csv data to produce a HIDECAN plot.

Usage

```
run_hidecan_shiny()
```

Value

No return value, called for side effects (launching the shiny app).

validate_CAN_data	<i>Checks validity of input for CAN_data constructor</i>
-------------------	--

Description

Checks validity of input for CAN_data constructor

Usage

```
validate_CAN_data(x)
```

Arguments

x a CAN_data object constructed via [new_CAN_data](#).

Value

A CAN_data object, i.e. a tibble.

validate_DE_data	<i>Checks validity of input for DE_data constructor</i>
------------------	---

Description

Checks validity of input for DE_data constructor

Usage

```
validate_DE_data(x)
```

Arguments

x a DE_data object constructed via [new_DE_data](#).

Value

A DE_data object, i.e. a tibble.

validate_GWAS_data	<i>Checks validity of input for GWAS_data constructor</i>
--------------------	---

Description

Checks validity of input for GWAS_data constructor

Usage

```
validate_GWAS_data(x)
```

Arguments

x a GWAS_data object constructed via [new_GWAS_data](#).

Value

A GWAS_data object, i.e. a tibble.

Index

[.check_cols](#), [2](#)
[.compute_chrom_length_genes](#), [3](#)
[apply_threshold](#), [3](#), [7](#)
[CAN_data](#), [4](#)
[combine_chrom_length](#), [5](#), [7](#)
[compute_chrom_length](#), [6](#)
[create_hidecan_plot](#), [6](#), [15](#)
[DE_data](#), [9](#)
[geom_label_repel](#), [8](#), [13](#)
[get_example_data](#), [10](#)
[GWAS_data](#), [10](#)
[GWAS_data_from_gwaspoly](#), [11](#)
[hidecan_plot](#), [12](#)
[hidecan_plot_from_gwaspoly](#), [15](#)
[new_CAN_data](#), [15](#), [17](#)
[new_DE_data](#), [16](#), [18](#)
[new_GWAS_data](#), [16](#), [18](#)
[run_hidecan_shiny](#), [17](#)
[validate_CAN_data](#), [17](#)
[validate_DE_data](#), [18](#)
[validate_GWAS_data](#), [18](#)