Package ‘tidyHeatmap’

June 23, 2020

Type Package

Title A Tidy Implementation of Heatmap

Version 1.0.1

Maintainer Stefano Mangiola <mangiolastefano@gmail.com>

Description This is a tidy implementation for heatmap. At the moment it is based on the (great) package ‘ComplexHeatmap’. The goal of this package is to interface a tidy data frame with this powerful tool. Some of the advantages are: Row and/or columns colour annotations are easy to integrate just specifying one parameter (column names). Custom grouping of rows is easy to specify providing a grouped tbl. For example: df %>% group_by(...). Labels size adjusted by row and column total number. Default use of Brewer and Viridis palettes.

License GPL-3

URL https://www.r-project.org,
    https://github.com/stemangiola/tidyHeatmap

BugReports https://github.com/stemangiola/tidyHeatmap

Depends R (>= 3.6),
    ComplexHeatmap (>= 2.2.0)

Imports methods,
    stats,
    utils,
    dplyr (>= 0.8.5),
    magrittr (>= 1.5),
    tidyr (>= 1.0.3),
    rlang (>= 0.4.5),
    purrr (>= 0.3.3),
    viridis (>= 0.5.1),
    circlize (>= 0.4.8),
    RColorBrewer (>= 1.1),
    grid,
    grDevices,
    lifecycle (>= 0.2.0)

Suggests testthat,
    BiocManager,
    knitr,
R topics documented:

- add_attr
- add_class
- annot_to_list
- as_matrix
- check_if_counts_is_na
- check_if_duplicated_genes
- check_if_wrong_input
- drop_class
- error_if_log_transformed
- get_abundance_norm_if_exists
- get_elements
- get_elements_features
- get_elements_features_abundance
- get_sample_counts
- get_sample_transcript
- get_sample_transcript_counts
- get_x_y_annotation_columns
- heatmap
- heatmap.default
- heatmap.tbl_df
- ifelse2_pipe
- ifelse_pipe
- N52
- parse_formula
- pasilla
- plot_heatmap
- prepend
- quo_names
- save_pdf
- save_pdf,Heatmap-method
- scale_design
- scale_robust
- select_closest_pairs
- type_to_annot_function

Index 24
add_attr

Description
Add attribute to abject

Usage
add_attr(var, attribute, name)

Arguments
var A tibble
attribute An object
name A character name of the attribute

Value
A tibble with an additional attribute

add_class

Description
Add class to abject

Usage
add_class(var, name)

Arguments
var A tibble
name A character name of the attribute

Value
A tibble with an additional attribute
annot_to_list

Description
annot_to_list

Usage
annot_to_list(.data)

Arguments
.data A data frame

Value
A list

as_matrix

Get matrix from tibble

Description
Get matrix from tibble

Usage
as_matrix(tbl, rownames = NULL, do_check = TRUE)

Arguments
tbl A tibble
rownames A character string of the rownames
do_check A boolean

Value
A matrix
check_if_counts_is_na  

Check whether there are NA counts

Description

Check whether there are NA counts

Usage

check_if_counts_is_na(.data, .abundance)

Arguments

.data  A tibble of read counts
.abundance  A character name of the read count column

Value

A tbl

check_if_duplicated_genes

Check whether there are duplicated genes/transcripts

Description

Check whether there are duplicated genes/transcripts

Usage

check_if_duplicated_genes(.data, .sample, .transcript, .abundance)

Arguments

.data  A tibble of read counts
.sample  A character name of the sample column
.transcript  A character name of the transcript/gene column
.abundance  A character name of the read count column

Value

A tbl
check_if_wrong_input  

*Check whether there are NA counts*

**Description**

Check whether there are NA counts

**Usage**

`check_if_wrong_input(.data, list_input, expected_type)`

**Arguments**

- **.data**  
  A tibble of read counts
- **list_input**  
  A list
- **expected_type**  
  A character string

**Value**

A tbl

---

drop_class  

*Remove class to abject*

**Description**

Remove class to abject

**Usage**

`drop_class(var, name)`

**Arguments**

- **var**  
  A tibble
- **name**  
  A character name of the class

**Value**

A tibble with an additional attribute
error_if_log_transformed

Check whether a numeric vector has been log transformed

Description
Check whether a numeric vector has been log transformed

Usage
error_if_log_transformed(x, .abundance)

Arguments
x A numeric vector
.abundance A character name of the transcript/gene abundance column

Value
NA

get_abundance_norm_if_exists
Get column names either from user or from attributes

Description
Get column names either from user or from attributes

Usage
get_abundance_norm_if_exists(.data, .abundance)

Arguments
.data A tibble
.abundance A character name of the abundance column

Value
A list of column enquo or error
get_elements  
*Get column names either from user or from attributes*

**Description**

Get column names either from user or from attributes

**Usage**

```r
get_elements(.data, .element, of_samples = TRUE)
```

**Arguments**

- `.data`  
  A tibble

- `.element`  
  A character name of the sample column

- `.of_samples`  
  A boolean

**Value**

A list of column enquo or error

---

get_elements_features  
*Get column names either from user or from attributes*

**Description**

Get column names either from user or from attributes

**Usage**

```r
get_elements_features(.data, .element, .feature, of_samples = TRUE)
```

**Arguments**

- `.data`  
  A tibble

- `.element`  
  A character name of the sample column

- `.feature`  
  A character name of the transcript/gene column

- `.of_samples`  
  A boolean

**Value**

A list of column enquo or error
**get_elements_features_abundance**

Get column names either from user or from attributes

**Usage**

```r
get_elements_features_abundance(
  .data, 
  .element, 
  .feature, 
  .abundance, 
  of_samples = TRUE 
)
```

**Arguments**

- `.data` A tibble
- `.element` A character name of the sample column
- `.feature` A character name of the transcript/gene column
- `.abundance` A character name of the read count column
- `of_samples` A boolean

**Value**

A list of column enquo or error

---

**get_sample_counts**

Get column names either from user or from attributes

**Usage**

```r
get_sample_counts(.data, .sample, .abundance)
```

**Arguments**

- `.data` A tibble
- `.sample` A character name of the sample column
- `.abundance` A character name of the read count column

**Value**

A list of column enquo or error
**get_sample_transcript**  
*Get column names either from user or from attributes*

**Description**
Get column names either from user or from attributes

**Usage**
```
get_sample_transcript(.data, .sample, .transcript)
```

**Arguments**
- `.data` A tibble
- `.sample` A character name of the sample column
- `.transcript` A character name of the transcript/gene column

**Value**
A list of column enquo or error

**get_sample_transcript_counts**
*Get column names either from user or from attributes*

**Description**
Get column names either from user or from attributes

**Usage**
```
get_sample_transcript_counts(.data, .sample, .transcript, .abundance)
```

**Arguments**
- `.data` A tibble
- `.sample` A character name of the sample column
- `.transcript` A character name of the transcript/gene column
- `.abundance` A character name of the read count column

**Value**
A list of column enquo or error
Description

get_x_y_annotation_columns

Usage

get_x_y_annotation_columns(.data, .column, .row, .abundance)

Arguments

.data A 'tbl' formatted as | <SAMPLE> | <TRANSCRIPT> | <COUNT> | <...>
.column The name of the column horizontally presented in the heatmap
.row The name of the column vertically presented in the heatmap
.abundance The name of the transcript/gene abundance column

Value

A list

heatmap

Creates a 'ComplexHeatmap' plot from 'tbl_df'

Description

heatmap() takes a tbl object and easily produces a ComplexHeatmap plot, with integration with tibble and dplyr frameworks.

Usage

heatmap(
   .data,
   .row,
   .column,
   .value,
   annotation = NULL,
   type = rep("tile", length(quo_names(annotation))),
   transform = NULL,
   .scale = "row",
   palette_value = c("#440154FF", "#21908CFF", "#fefada"),
   palette_discrete = list(),
   palette_continuous = list(),
   .abundance = NULL,
   .horizontal = NULL,
   .vertical = NULL,
   log_transform = NULL,
palette_abundance = NULL,
...
)

Arguments

.data A `tbl_df` formatted as | <ELEMENT> | <FEATURE> | <VALUE> | <...> |
.row The name of the column vertically presented in the heatmap
.column The name of the column horizontally presented in the heatmap
.value The name of the transcript/gene abundance column
.annotation Vector of quotes
.type A character vector of the set c("tile", "point", "bar", "line")
.transform A function, used to transform .value row-wise (e.g., transform = log1p)
.scale A character string. Possible values are c("none", "row", "column", "both")
palette_value A character vector This is the palette that will be used as gradient for .value
palette_discrete A list of character vectors. This is the list of palettes that will be used for horizontal and vertical discrete annotations. The discrete classification of annotations depends on the column type of your input tibble (e.g., character and factor).
palette_continuous A list of character vectors. This is the list of palettes that will be used for horizontal and vertical continuous annotations. The continuous classification of annotations depends on the column type of your input tibble (e.g., integer, numerical, double).
.abundance DEPRECATED. Please use .value instead
.horizontal DEPRECATED. Please use .column instead
.vertical DEPRECATED. Please use .row instead
.log_transform DEPRECATED. Please use transform instead
.palette_abundance DEPRECATED. Please use palette_value instead
...
Further arguments to be passed to ComplexHeatmap::Heatmap

Details

Maturing
To be added.

Value

A `ComplexHeatmap` object

Examples

library(dplyr)
tidyHeatmap::N52 %>%
group_by( `Cell type`) %>%
tidyHeatmap::heatmap(
    .row = symbol_ct,
heatmap.default

/grave.Var
read count normalised log
/grave.Var

heatmap.default

## Default S3 method:
heatmap(
  .data,
  .row,
  .column,
  .value,
  annotation = NULL,
  type = rep("tile", length(quo_names(annotation))),
  transform = NULL,
  .scale = "row",
  palette.value = c("#440154FF", "#21908CFF", "#efada"),
  palette.discrete = list(),
  palette.continuous = list(),
  .abundance = NULL,
  .horizontal = NULL,
  .vertical = NULL,
  log.transform = NULL,
  palette.abundance = NULL,
  ...)

Arguments

.data A `tbl_df` formatted as |

.row The name of the column vertically presented in the heatmap |

.column The name of the column horizontally presented in the heatmap |

.value The name of the transcript/gene abundance column |

.annotation Vector of quotes |

type A character vector of the set c("tile", "point", "bar", "line") |

.transform A function, used to transform .value row-wise (e.g., transform = log1p) |

.scale A character string. Possible values are c("none", "row", "column", "both") |

.palette.value A character vector This is the palette that will be used as gradient for .value |
palette_discrete
A list of character vectors. This is the list of palettes that will be used for horizontal and vertical discrete annotations. The discrete classification of annotations depends on the column type of your input tibble (e.g., character and factor).

palette_continuous
A list of character vectors. This is the list of palettes that will be used for horizontal and vertical continuous annotations. The continuous classification of annotations depends on the column type of your input tibble (e.g., integer, numerical, double).

.abundance DEPRECATED. Please use .value instead
.horizontal DEPRECATED. Please use .column instead
.vertical DEPRECATED. Please use .row instead
.log_transform DEPRECATED. Please use transform instead
.palette_abundance
DEPRECATED. Please use palette_value instead

... Further arguments to be passed to ComplexHeatmap::Heatmap

---

heatmap.tbl_df,

Creates a ‘ComplexHeatmap’ plot from ‘tbl_df’

Description
Creates a ‘ComplexHeatmap’ plot from ‘tbl_df’

Usage

## S3 method for class 'tbl_df'
heatmap(
  .data,
  .row,
  .column,
  .value,
  annotation = NULL,
  type = rep("tile", length(quo_names(annotation))),
  transform = NULL,
  .scale = "row",
  palette_value = c("#440154FF", "#21908CFF", "#fefada"),
  palette_discrete = list(),
  palette_continuous = list(),
  .abundance = NULL,
  .horizontal = NULL,
  .vertical = NULL,
  log_transform = NULL,
  palette_transform = NULL,
  palette_abundance = NULL,
  ...
)
Arguments

.data A 'tbl_df' formatted as | <ELEMENT> | <FEATURE> | <VALUE> | <...> |
.row The name of the column vertically presented in the heatmap
.column The name of the column horizontally presented in the heatmap
.value The name of the transcript/gene abundance column
.annotation Vector of quotes
.type A character vector of the set c("tile", "point", "bar", "line")
.transform A function, used to transform .value row-wise (e.g., transform = log1p)
.scale A character string. Possible values are c("none", "row", "column", "both")
palette_value A character vector. This is the palette that will be used as gradient for .value
palette_discrete A list of character vectors. This is the list of palettes that will be used for horizontal and vertical discrete annotations. The discrete classification of annotations depends on the column type of your input tibble (e.g., character and factor).
palette_continuous A list of character vectors. This is the list of palettes that will be used for horizontal and vertical continuous annotations. The continuous classification of annotations depends on the column type of your input tibble (e.g., integer, numerical, double).
.abundance DEPRECATED. Please use .value instead
.horizontal DEPRECATED. Please use .column instead
.vertical DEPRECATED. Please use .row instead
.log_transform DEPRECATED. Please use transform instead
.palette_abundance DEPRECATED. Please use palette_value instead
...
Further arguments to be passed to ComplexHeatmap::Heatmap

ifelse2_pipe

This is a generalisation of ifelse that accepts an object and return an objects

Description

This is a generalisation of ifelse that accepts an object and return an objects

Usage

ifelse2_pipe(.x, .p1, .p2, .f1, .f2, .f3 = NULL)

Arguments

.x A tibble
.p1 A boolean
.p2 ELSE IF condition
.f1 A function
.f2 A function
.f3 A function
Value

A tibble

ifelse_pipe

This is a generalisation of ifelse that accepts an object and return an objects

Description

This is a generalisation of ifelse that accepts an object and return an objects

Usage

ifelse_pipe(.x, .p, .f1, .f2 = NULL)

Arguments

\[.x\] A tibble
\[.p\] A boolean
\[.f1\] A function
\[.f2\] A function

Value

A tibble

N52

Example data set N52

Description

Example data set N52

Usage

N52

Format

An object of class tbl_df (inherits from tbl_data.frame) with 520 rows and 15 columns.
parse_formula

Description
	.formula parser

Usage
 parse_formula(fm)

Arguments
 fm 
a formula

Value
 A character vector

pasilla

Description
 Example data set Pasilla

Usage
 pasilla

Format
 An object of class tbl_df (inherits from tbl, data.frame) with 504 rows and 7 columns.

plot_heatmap

Description
 plot_heatmap() takes a tbl object and easily produces a ComplexHeatmap plot, with integration with tibble and dplyr frameworks.
Usage

```r
plot_heatmap(
  .data,  
  .horizontal,  
  .vertical,  
  .abundance,  
  annotation = NULL,  
  type = rep("tile", length(quo_names(annotation))),  
  transform = NULL,  
  .scale = "row",  
  palette_abundance = c("#440154FF", "#21908CFF", "#fefada"),  
  palette_discrete = list(),  
  palette_continuous = list(),
  ...
)
```

Arguments

- `.data`  
  A `tbl` formatted as `| <SAMPLE> | <TRANSCRIPT> | <COUNT> | <...> |`

- `.horizontal`  
  The name of the column horizontally presented in the heatmap

- `.vertical`  
  The name of the column vertically presented in the heatmap

- `.abundance`  
  The name of the transcript/gene abundance column

- `annotation`  
  Vector of quotes

- `type`  
  A character vector of the set `c("tile", "point", "bar", "line")`

- `transform`  
  A function, used to transform `.value`, for example `log1p`

- `.scale`  
  A character string. Possible values are `c("none", "row", "column", "both")`

- `palette_abundance`  
  A character vector, or a function for higher customisation (colorRamp2). This is the palette that will be used as gradient for abundance. If `palette_abundance` is a vector of hexadecimal colous, it should have 3 values. If you want more customisation, you can pass to `palette_abundance` a function, that is derived as for example `colorRamp2(c(-2, 0, 2), palette_abundance)`

- `palette_discrete`  
  A list of character vectors. This is the list of palettes that will be used for horizontal and vertical discrete annotations. The discrete classification of annotations depends on the column type of your input tibble (e.g., character and factor).

- `palette_continuous`  
  A list of character vectors. This is the list of palettes that will be used for horizontal and vertical continuous annotations. The continuous classification of annotations depends on the column type of your input tibble (e.g., integer, numerical, double).

- `...`  
  Further arguments to be passed to `ComplexHeatmap::Heatmap`

Details

To be added.

Value

A `ComplexHeatmap` object
**prepend**

*From rlang deprecated*

---

**Description**

From rlang deprecated

**Usage**

`prepend(x, values, before = 1)`

**Arguments**

- `x`: An array
- `values`: An array
- `before`: A boolean

**Value**

An array

---

**quo_names**

*Convert array of quosure (e.g. c(col_a, col_b)) into character vector*

---

**Description**

Convert array of quosure (e.g. c(col_a, col_b)) into character vector

**Usage**

`quo_names(v)`

**Arguments**

- `v`: A array of quosures (e.g. c(col_a, col_b))

**Value**

A character vector
**Description**

save_pdf() takes as input a Heatmap from ComplexHeatmap and save it to PDF file

**Usage**

```r
save_pdf(
  .heatmap,
  filename,
  width = NULL,
  height = NULL,
  units = c("in", "cm", "mm")
)
```

**Arguments**

- `.heatmap` A 'Heatmap'
- `.filename` A character string. The name of the output file/path
- `.width` A 'double'. Plot width
- `.height` A 'double'. Plot height
- `.units` A character string. units ("in", "cm", or "mm")

**Details**

**Maturing**

It simply save an 'Heatmap' to a PDF file use pdf() function in the back end

**Value**

NA

**Examples**

```r
tidyHeatmap::heatmap(
  dplyr::group_by(tidyHeatmap::pasilla,location, type),
  .column = sample,
  .row = symbol,
  .value = \`count normalised adjusted`,
) `>%`
save_pdf(tempfile())
```
save_pdf

Description
save_pdf

Usage
## S4 method for signature 'Heatmap'
save_pdf(
  .heatmap,
  filename,
  width = NULL,
  height = NULL,
  units = c("in", "cm", "mm")
)

Arguments

  .heatmap    A 'Heatmap'
  filename    A character string. The name of the output file/path
  width       A 'double'. Plot width
  height      A 'double'. Plot height
  units       A character string. units ("in", "cm", or "mm")

scale_design

Description
Scale design matrix

Usage
  scale_design(df, .formula)

Arguments

  df         A tibble
  .formula   a formula

Value
A tibble
scale_robust

Scale counts in a robust way against sd == 0

Description

Scale counts in a robust way against sd == 0

Usage

scale_robust(y)

Arguments

y A numerical array

Value

A scaled and centred numerical array

select_closest_pairs

Sub function of remove_redundancy_elements_though_reduced_dimensions

Description

Sub function of remove_redundancy_elements_though_reduced_dimensions

Usage

select_closest_pairs(df)

Arguments

df A tibble

Value

A tibble with pairs to drop
type_to_annot_function

Description

Usage

Format

An object of class list of length 4.
Index

*Topic datasets
  N52, 16
  pasilla, 17
  type_to_annot_function, 23

add_attr, 3
add_class, 3
annot_to_list, 4
as_matrix, 4

check_if_counts_is_na, 5
check_if_duplicated_genes, 5
check_if_wrong_input, 6
drop_class, 6
error_if_log_transformed, 7
get_abundance_norm_if_exists, 7
get_elements, 8
get_elements_features, 8
get_elements_features_abundance, 9
get_sample_counts, 9
get_sample_transcript, 10
get_sample_transcript_counts, 10
get_x_y_annotation_columns, 11

heatmap, 11
heatmap.default, 13
heatmap.tbl_df, 14

ifelse2_pipe, 15
ifelse_pipe, 16

N52, 16

parse_formula, 17
pasilla, 17
plot_heatmap, 17
prepend, 19

quo_names, 19

save_pdf, 20
save_pdf, Heatmap-method, 21
scale_design, 21
scale_robust, 22
select_closest_pairs, 22
type_to_annot_function, 23