Package ‘HarmonizR’

February 28, 2024

**Title** Handles missing values and makes more data available

**Version** 1.0.0

**Description** An implementation, which takes input data and makes it available for proper batch effect removal by ComBat or Limma. The implementation appropriately handles missing values by dissecting the input matrix into smaller matrices with sufficient data to feed the ComBat or limma algorithm. The adjusted data is returned to the user as a rebuild matrix. The implementation is meant to make as much data available as possible with minimal data loss.

**Depends** R (>= 4.2.0)

**License** GPL-3

**Encoding** UTF-8

**Roxygen** list(markdown = TRUE)

**RoxygenNote** 7.2.3

**biocViews** BatchEffect

**Imports** doParallel (>= 1.0.16), foreach (>= 1.5.1), janitor (>= 2.1.0), plyr (>= 1.8.6), sva (>= 3.36.0), seriation (>= 1.3.5), limma (>= 3.46.0), SummarizedExperiment

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

**VignetteBuilder** knitr

**Config/testthat/edition** 3

**git_url** https://git.bioconductor.org/packages/HarmonizR

**git_branch** RELEASE_3_18

**git_last_commit** 6404f6c

**git_last_commit_date** 2023-10-24

**Repository** Bioconductor 3.18

**Date/Publication** 2024-02-28

**Author** Simon Schlumbohm [aut, cre], Julia Neumann [aut], Philipp Neumann [aut]

**Maintainer** Simon Schlumbohm <schlumbohm@hsu-hh.de>
**R topics documented:**

- binary_matrix_reduction .......................... 2
- blocking ........................................... 3
- build_key_list .................................... 3
- fetch_batch_overview .............................. 4
- find_na ........................................... 4
- format_from_S4 .................................... 5
- format_to_s4 ...................................... 5
- harmonizR .......................................... 6
- jaccard ............................................ 7
- jaccard_index_absence ............................. 8
- jaccard_index_existence ........................... 8
- read_description .................................. 9
- read_main_data .................................... 9
- rebuild ........................................... 10
- sorting ........................................... 10
- splitting ......................................... 11
- spotting_missing_values .......................... 12
- unique_removal ................................... 12
- visual ............................................ 13
- visual2 ........................................... 13
- visual3 ........................................... 14

**Index**

---

**binary_matrix_reduction**

*Creating a binary existence matrix*

### Description

This function reduces its input matrix to a binary existence matrix based on the given description file (and information on how many values a batch needs) for proper adjustment.

### Usage

```r
binary_matrix_reduction(binary_data, batch_list, needed_values)
```

### Arguments

- **binary_data** The input data.frame that should become binary.
- **batch_list** Information about the sample’s batch affiliations.
- **needed_values** Information, how many values are needed to render a a batch 'valid'.

### Value

A binary existence matrix returned as a data.frame
blocking

blocking

blocking

blocking

Description
This function performs blocking on the given description and therefore influences how the dataset will be split later down the pipeline.

Usage
blocking(batch_list, block)

Arguments
batch_list The list with information about batch-affiliations for every sample.
block The blocking parameter (how many batches should always get blocked together).

Value
Returns an updated 'batch_list' with blocking included

build_key_list

build_key_list

Description
Calculates a list of usable keys based on the passed batch listings

Usage
build_key_list(batch_list)

Arguments
batch_list The list with information about batch-affiliations for every sample.

Value
A list element with usable keys
fetch_batch_overview

*Fetching batch list*

**Description**

The `fetch_batch_overview` function extracts the overview over the batch distribution in list format.

**Usage**

```
fetch_batch_overview(batch_data)
```

**Arguments**

- `batch_data`: This is a `data.frame` and simultaneously the result from `read_description()`

**Value**

Batch distribution as list

---

find_na

*Finding NAs for the sorting process*

**Description**

Creates an overview of NAs based on both the passed input `data.frame` and the batch list.

**Usage**

```
find_na(df, batch_list)
```

**Arguments**

- `df`: The `data.frame` passed initially by the user.
- `batch_list`: The list with information about batch-affiliations for every sample.

**Value**

An overview of the NA-distribution
**format_from_S4**

**Format data taken from S4**

**Description**

This function converts passed S4 summarized experiment data to HarmonizR input

**Usage**

```r
format_from_S4(data)
```

**Arguments**

- `data` (S4 format) passed by the user. No description file is needed when using S4 data

**Value**

Data and description as data.frames

---

**format_to_s4**

**Format data taken from HarmonizR back to S4**

**Description**

This function converts passed HarmonizR output to a S4 summarized experiment data structure

**Usage**

```r
format_to_s4(cured_data, s4_saved)
```

**Arguments**

- `cured_data` The HarmonizR output
- `s4_saved` The original S4 input

**Value**

The HarmonizR output formatted as S4 data
harmonizR  

Main function

Description

This function executes the entire HarmonizR program and executes all other functions found in this package. Therefore, this is the only function in need of calling.

Usage

```r
harmonizR(
  data_as_input = NULL,
  description_as_input = NULL,
  ...,  
  algorithm = "ComBat",
  ComBat_mode = 1,
  plot = FALSE,
  sort = FALSE,
  block = NULL,
  output_file = "cured_data",
  verbosity = 1,
  cores = FALSE,
  ur = TRUE
)
```

Arguments

- **data_as_input**: Path to input data. Additionally, the input can be a data.frame with proper row-and column names.
- **description_as_input**: Path to input description. Additionally, the input can be a data.frame with three columns total.
- **...**: Unsettable parameter. Used to make all parameters below optional. Documented to adhere with Bioconductor guidelines.
- **algorithm**: Optional. Pass either "ComBat" or "limma" to select the preferred adjustment method. Defaults to ComBat.
- **ComBat_mode**: Optional. Pass a number between 1 and 4 to select the desired ComBat parameters. Can only be set when ComBat is used. For information on the meaning of the numbers, please view the SOP. Defaults to 1.
- **plot**: Optional. Takes either "samplemeans" for sample specific means, "feature-means" for feature specific means or "CV" for the coefficient of variation as input and creates before/after plots for the given data. When set, additionally writes out a .pdf file. Defaults to FALSE -> Turned off.
- **sort**: Optional. Method to sort by. Either FALSE or "sparsity_sort", "seriation_sort" or "jaccard_sort".
block

Optional. How many batches should be treated as one during blocking. Greatly affects the number of sub-dataframes produced and reduces runtime. Turned off by default.

output_file

Optional. Takes a string as input for the .tsv file name. This can also be a path. Defaults to "cured_data", hence yielding a "cured_data.tsv" file in the work directory from which it was called. Can be turned of by passing FALSE.

verbosity

Optional. Toggles the amount of information printed out by the HarmonizR algorithm during execution. Takes a number from 0 (also "mute") to any positive number. The higher, the more information will be printed. For the standard user, anything above 2 is rarely needed. Defaults to 1.

cores

Optional. Manually sets the number of cores the user wants to be used during HarmonizR's execution. Takes a positive integer. Defaults to the amount of available cores.

ur

Optional. Toggles the functionality of the removal of unique combinations for increased data rescue. Defaults to TRUE. Not recommended to set to FALSE, as it exists for testing and reproducibility purposes.

Value

The batch effect adjusted data.frame. Additionally, a .tsv file by default called "cured_data.tsv" will be written out as a result

Examples

```r
# create a dataframe with 3 rows and 6 columns filled with random numbers
df <- data.frame(matrix(rnorm(n = 3*6), ncol = 6))
# set the column names
colnames(df) <- c("A", "B", "C", "D", "E", "F")
# create a vector of row names
row_names <- c("F1", "F2", "F3")
# set the row names
rownames(df) <- row_names

# create a vector of batch numbers
batch <- rep(1:3, each = 2)
# create a dataframe with 6 rows and 3 columns
des <- data.frame(ID = colnames(df), sample = 1:6, batch = batch)

# use the harmonizR() function; turning off creation of an output .tsv file
harmonizR(df, des, output_file = FALSE, cores = 1)
```

---

**jaccard**

Jaccard-based sorting

**Description**

Calculates a order to sort by based on the Jaccard similarity of all given batches
Usage

jaccard(binary_df)

Arguments

binary_df  The input matrix passed by the user reduced to presence and absence of features in batches (binary)

Value

A template for batch-sorting based on Jaccard similarity

jaccard_index_absence  Jaccard index on zeroes (absence)

Description

Calculates the Jaccard index for two given lists a and b based on common zeroes

Usage

jaccard_index_absence(a, b)

Arguments

a  First list with either 0 or 1 entries to be compared against the second list.
b  Second list with either 0 or 1 entries to be compared against the first list.

Value

The Jaccard similarity based on absent values

jaccard_index_existence  Jaccard index on ones (existence)

Description

Calculates the Jaccard index for two given lists a and b based on common ones

Usage

jaccard_index_existence(a, b)
### read_description

**Arguments**
- **a**  
  First list with either 0 or 1 entries to be compared against the second list.
- **b**  
  Second list with either 0 or 1 entries to be compared against the first list.

**Value**

The Jaccard similarity based on existing values

---

#### Description

The `read_description` function reads in a file via its file path and converts it to a for the rest of the workflow readable format.

#### Usage

```r
read_description(description_source)
```

**Arguments**
- **description_source**  
  Usually the path to the description file. It can also be a correctly formatted data.frame.

**Value**

Description as data.frame

---

#### read_main_data

**Description**

The `read_main_data` function reads in a file via its file path and converts it to a for the rest of the workflow readable format.

**Usage**

```r
read_main_data(data_source)
```

**Arguments**
- **data_source**  
  Usually the path to the input data. It can also be passed directly as a correctly formatted data.frame.
Value
To-be-adjusted data as data.frame

rebuild Rebuilding

Description
The rebuild function rebuilds the sub-dataframes to one big output data.frame.

Usage
rebuild(cured_subdfs)

Arguments
cured_subdfs a list of data.frames, which are the result from splitting().

Value
The rebuild() function returns the adjusted data.frame and writes out cured_data.tsv

sorting Sorting the input data.frame

Description
Creates an overview of NAs based on both the passed input data.frame and the batch list.

Usage
sorting(df, batch_list, batch_data, order_to_go_by, verbosity)

Arguments
df The data.frame passed initially by the user.
batch_list The list with information about batch-affiliations for every sample.
batch_data The full data.frame passed as description by the user.
order_to_go_by The template to sort by.
verbatim Toggles the amount of information printed out by the HarmonizR algorithm during execution. Passed on from the main function.

Value
Correctly sorted data and description as two elements of a list
splitting

Description

This function splits the data.frame. The data is very sensitive to its specific input. Only to be called via harmonizR()

Usage

splitting(
  affiliation_list,
  main_data,
  batch_data,
  block_list,
  algorithm,
  ComBat_mode,
  block,
  verbosity,
  cores
)

Arguments

affiliation_list An overview of which protein has which missing value distribution.
main_data This is the input data.frame read in by the HarmonizR.
batch_data This is the description data.frame read in by the HarmonizR.
block_list An overview of the batch groupings in list form. If the block parameter was used, the groupings are changed accordingly.
algorithm Either "ComBat" or "limma". Based on the selected algorithm for the harmonizR() function. Not active during limma execution.
ComBat_mode The chosen ComBat mode influences the parameters the ComBat algorithm is using. Based on the ComBat_mode parameter given to the harmonizR() function. Not active during limma execution.
block The block parameter is here used to determine whether there are single-batch dataframes at all present.
verbosity Toggles the amount of stuff printed out by the HarmonizR algorithm during execution.
cores Manually sets the number of cores the user wants to be used during HarmonizR’s execution. A positive integer.

Value

Returns a list of `chopped up` data.frames
spoting_missing_values

*Spotting*

**Description**

This function spots missing values within the given data.frame.

**Usage**

```r
spoting_missing_values(
  main_data,
  batch_list,
  block_list,
  needed_values,
  verbosity
)
```

**Arguments**

- `main_data`: This is the input data.frame read in by the HarmonizR.
- `batch_list`: An overview of the batch groupings in list form (comes from the user).
- `block_list`: An overview of the batch groupings in list form (comes from the blocking function). If blocking is FALSE, this list will be the same as 'batch_list'.
- `needed_values`: The number of values needed to be present in a batch in order to be valid.
- `verbosity`: Toggles the amount of stuff printed out by the HarmonizR algorithm during execution.

**Value**

A list of vectors to pass to the upcoming splitting() function.

---

**unique_removal**

*Remove unique combinations*

**Description**

The unique_removal function changes the gathered information of the features in a way that guarantees no single-line sub-dataframes to appear, causing less data loss.

**Usage**

```r
unique_removal(affiliation_list)
```
Arguments
   affiliation_list
      An overview of which protein has which missing value distribution.

Value
   Updated version of the passed affiliation_list

visual
   Visualize feature means

Description
   The visual functions turn their input dataframes into easily plottable results.

Usage
   visual(input_dataframe, batch_list)

Arguments
   input_dataframe
      A data.frame object as input.
   batch_list
      A list object giving information about which column corresponds to which batch.

Value
   A data.frame object, which is ready to be plotted

visual2
   Visualize sample means

Description
   The visual functions turn their input dataframes into easily plottable results.

Usage
   visual2(input_dataframe, batch_list)

Arguments
   input_dataframe
      A data.frame object as input.
   batch_list
      A list object giving information about which column corresponds to which batch.
**Value**

A data.frame object, which is ready to be plotted

---

**visual3**

*Visualize CV*

---

**Description**

The visual functions turn their input dataframes into easily plottable results.

**Usage**

`visual3(input_dataframe, batch_list)`

**Arguments**

- `input_dataframe`  
  A data.frame object as input.

- `batch_list`  
  A list object giving information about which column corresponds to which batch.

**Value**

A data.frame object, which is ready to be plotted
Index

binary_matrix_reduction, 2
blocking, 3
build_key_list, 3
fetch_batch_overview, 4
find_na, 4
format_from_S4, 5
format_to_s4, 5
harmonizR, 6
jaccard, 7
jaccard_index_absence, 8
jaccard_index_existence, 8
read_description, 9
read_main_data, 9
rebuild, 10
sorting, 10
splitting, 11
spotting_missing_values, 12
unique_removal, 12
visual, 13
visual2, 13
visual3, 14