

Package ‘NACHO’

July 21, 2025

Type Package

Title NanoString Quality Control Dashboard

Version 2.0.6

Description NanoString nCounter data are gene expression assays where there is no need for the use of enzymes or amplification protocols and work with fluorescent barcodes (Geiss et al. (2018) <[doi:10.1038/nbt1385](https://doi.org/10.1038/nbt1385)>). Each barcode is assigned a messenger-RNA/micro-RNA (mRNA/miRNA) which after bonding with its target can be counted. As a result each count of a specific barcode represents the presence of its target mRNA/miRNA. 'NACHO' (NAnoString quality Control dasHbOard) is able to analyse the exported NanoString nCounter data and facilitates the user in performing a quality control. 'NACHO' does this by visualising quality control metrics, expression of control genes, principal components and sample specific size factors in an interactive web application.

License GPL-3

URL <https://github.com/mcanouil/NACHO/>, <https://m.canouil.dev/NACHO/>

BugReports <https://github.com/mcanouil/NACHO/issues>

Depends R (>= 3.6.0)

Imports utils, data.table, ggplot2 (>= 3.3.0), ggforce (>= 0.3.1),
ggrepel (>= 0.8.1), knitr (>= 1.25), rmarkdown (>= 1.16), shiny
(>= 1.4.0), shinyWidgets (>= 0.4.9)

Suggests roxygen2 (>= 7.2.0), testthat (>= 2.2.1), covr (>= 3.3.2),
Biobase, GEOquery, limma

VignetteBuilder knitr

Encoding UTF-8

LazyData true

RoxygenNote 7.3.0

SystemRequirements pandoc (>= 1.14), pandoc-citeproc

Config/testthat/edition 3

NeedsCompilation no

Author Mickaël Canouil [aut, cre] (ORCID: <https://orcid.org/0000-0002-3396-4549>),
Roderick Sliker [aut] (ORCID: <https://orcid.org/0000-0003-0961-9152>),
Gerard Bouland [aut]

Maintainer Mickaël Canouil <pro@mickael.canouil.dev>

Repository CRAN

Date/Publication 2024-01-12 19:10:02 UTC

Contents

autoplot.nacho	2
check_outliers	4
deploy	4
GSE74821	5
load_rcc	5
normalise	7
print.nacho	10
render	11
visualise	13
Index	15

autoplot.nacho	<i>Plot quality-control metrics and thresholds of a "nacho" object</i>
----------------	--

Description

This function allows to plot any qualit-control figures available within the shiny app using [visualise\(\)](#) or in the HTML report from [render\(\)](#).

Usage

```
## S3 method for class 'nacho'
autoplot(
  object,
  x,
  colour = "CartridgeID",
  size = 0.5,
  show_legend = TRUE,
  show_outliers = TRUE,
  outliers_factor = 1,
  outliers_labels = NULL,
  ...
)
```

Arguments

object	[list] List obtained from load_rcc() or normalise() .
x	[character] Character string naming the quality-control metrics to plot from nacho_object. The possible values are: • "BD" (Binding Density) • "FoV" (Imaging) • "PCL" (Positive Control Linearity) • "LoD" (Limit of Detection) • "Positive" (Positive Controls) • "Negative" (Negative Controls) • "Housekeeping" (Housekeeping Genes) • "PN" (Positive Controls vs. Negative Controls) • "ACBD" (Average Counts vs. Binding Density) • "ACMC" (Average Counts vs. Median Counts) • "PCA12" (Principal Component 1 vs. 2) • "PCAi" (Principal Component scree plot) • "PCA" (Principal Components planes) • "PFNF" (Positive Factor vs. Negative Factor) • "HF" (Housekeeping Factor) • "NORM" (Normalisation Factor)
colour	[character] Character string of the column in ssheet_csv or more generally in nacho_object\$nacho to be used as grouping colour.
size	[numeric] A numeric controlling point size (ggplot2::geom_point() or line size (ggplot2::geom_line()).
show_legend	[logical] Boolean to indicate whether the plot legends should be plotted (TRUE) or not (FALSE). Default is TRUE.
show_outliers	[logical] Boolean to indicate whether the outliers should be highlighted in red (TRUE) or not (FALSE). Default is TRUE.
outliers_factor	[numeric] Size factor for outliers compared to size. Default is 1.
outliers_labels	[character] Character to indicate which column in nacho_object\$nacho should be used to be printed as the labels for outliers or not. Default is NULL.
...	Other arguments (Not used).

Examples

```
data(GSE74821)

autoplot(GSE74821, x = "BD")
```

check_outliers	<i>Annotate a "nacho" object for outliers</i>
----------------	---

Description

Add or update "is_outlier" column in the "nacho" field of an object from a call to `load_rcc()` or `normalise()` (nacho_object\$nacho), using the current quality-control thresholds.

Usage

```
check_outliers(nacho_object)
```

Arguments

nacho_object [\[list\]](#) A list object of class "nacho" obtained from `load_rcc()` or `normalise()`.

Value

A [\[list\]](#) object of class "nacho".

Examples

```
data(GSE74821)
nacho_object <- check_outliers(GSE74821)
head(nacho_object$nacho)
```

deploy	<i>Deploy (copy) the shiny application to the specified directory</i>
--------	---

Description

Deploy (copy) the shiny application to the specified directory

Usage

```
deploy(directory = "/srv/shiny-server", app_name = "NACHO")
```

Arguments

directory [\[character\]](#) A character vector of one path to the new location.
 app_name [\[character\]](#) A character vector defining the shiny application name in the new location.

Value

[\[logical\]](#) A logical indicating whether the deployment is successfull (TRUE) or not (FALSE).

Examples

```

deploy(directory = ".")

if (interactive()) {
  shiny::runApp("NACHO")
}

```

GSE74821

*A "nacho" object containing 20 samples of GSE74821 dataset***Description**

NanoString nCounter RUO-PAM50 Gene Expression Custom CodeSet

Usage

```
GSE74821
```

Format

A [\[list\]](#) object of class "nacho".

Source

GSE74821

load_rcc

*Produce a "nacho" object from RCC NanoString files***Description**

This function is used to preprocess the data from NanoString nCounter.

Usage

```

load_rcc(
  data_directory,
  ssheet_csv,
  id_colname = NULL,
  housekeeping_genes = NULL,
  housekeeping_predict = FALSE,
  housekeeping_norm = TRUE,
  normalisation_method = "GEO",
  n_comp = 10
)

```

Arguments

- `data_directory` [[character](#)] A character string of the directory where the data are stored.
- `ssheet_csv` [[character](#)] or [[data.frame](#)] Either a string with the name of the CSV of the samplesheet or the samplesheet as a `data.frame`. Should contain a column that matches the file names in the folder.
- `id_colname` [[character](#)] Character string of the column in `ssheet_csv` that matches the file names in `data_directory`.
- `housekeeping_genes` [[character](#)] A vector of names of the miRNAs/mRNAs that should be used as housekeeping genes. Default is `NULL`.
- `housekeeping_predict` [[logical](#)] Boolean to indicate whether the housekeeping genes should be predicted (`TRUE`) or not (`FALSE`). Default is `FALSE`.
- `housekeeping_norm` [[logical](#)] Boolean to indicate whether the housekeeping normalisation should be performed. Default is `TRUE`.
- `normalisation_method` [[character](#)] Either "GEO" or "GLM". Character string to indicate normalisation using the geometric mean ("GEO") or a generalized linear model ("GLM"). Default is "GEO".
- `n_comp` [[numeric](#)] Number indicating the number of principal components to compute. Cannot be more than `n-1` samples. Default is 10.

Value

- [[list](#)] A list object of class "nacho":
- `access` [[character](#)] Value passed to `load_rcc()` in `id_colname`.
- `housekeeping_genes` [[character](#)] Value passed to `load_rcc()`.
- `housekeeping_predict` [[logical](#)] Value passed to `load_rcc()`.
- `housekeeping_norm` [[logical](#)] Value passed to `load_rcc()`.
- `normalisation_method` [[character](#)] Value passed to `load_rcc()`.
- `remove_outliers` [[logical](#)] `FALSE`.
- `n_comp` [[numeric](#)] Value passed to `load_rcc()`.
- `data_directory` [[character](#)] Value passed to `load_rcc()`.
- `pc_sum` [[data.frame](#)] A `data.frame` with `n_comp` rows and four columns: "Standard deviation", "Proportion of Variance", "Cumulative Proportion" and "PC".
- `nacho` [[data.frame](#)] A `data.frame` with all columns from the sample sheet `ssheet_csv` and all computed columns, *i.e.*, quality-control metrics and counts, with one sample per row.
- `outliers_thresholds` [[list](#)] A list of the (default) quality-control thresholds used.

Examples

```

if (interactive()) {
  library(GEOquery)
  library(NACHO)

  # Import data from GEO
  gse <- GEOquery::getGEO(GEO = "GSE74821")
  targets <- Biobase::pData(Biobase::phenoData(gse[[1]]))
  GEOquery::getGEOSuppFiles(GEO = "GSE74821", baseDir = tempdir())
  utils::untar(
    tarfile = file.path(tempdir(), "GSE74821", "GSE74821_RAW.tar"),
    exdir = file.path(tempdir(), "GSE74821")
  )
  targets$IDFILE <- list.files(
    path = file.path(tempdir(), "GSE74821"),
    pattern = ".RCC.gz$"
  )
  targets[] <- lapply(X = targets, FUN = iconv, from = "latin1", to = "ASCII")
  utils::write.csv(
    x = targets,
    file = file.path(tempdir(), "GSE74821", "Samplesheet.csv")
  )

  # Read RCC files and format
  nacho <- load_rcc(
    data_directory = file.path(tempdir(), "GSE74821"),
    ssheet_csv = file.path(tempdir(), "GSE74821", "Samplesheet.csv"),
    id_colname = "IDFILE"
  )
}

```

normalise

(re)Normalise a "nacho" object

Description

This function creates a list in which your settings, the raw counts and normalised counts are stored, using the result from a call to [load_rcc\(\)](#).

Usage

```

normalise(
  nacho_object,
  housekeeping_genes = nacho_object[["housekeeping_genes"]],
  housekeeping_predict = nacho_object[["housekeeping_predict"]],
  housekeeping_norm = nacho_object[["housekeeping_norm"]],
  normalisation_method = nacho_object[["normalisation_method"]],
  n_comp = nacho_object[["n_comp"]],

```

```

remove_outliers = nacho_object[["remove_outliers"]],
outliers_thresholds = nacho_object[["outliers_thresholds"]]
)

```

Arguments

nacho_object [list] A list object of class "nacho" obtained from `load_rcc()` or `normalise()`.

housekeeping_genes [character] A vector of names of the miRNAs/mRNAs that should be used as housekeeping genes. Default is NULL.

housekeeping_predict [logical] Boolean to indicate whether the housekeeping genes should be predicted (TRUE) or not (FALSE). Default is FALSE.

housekeeping_norm [logical] Boolean to indicate whether the housekeeping normalisation should be performed. Default is TRUE.

normalisation_method [character] Either "GEO" or "GLM". Character string to indicate normalisation using the geometric mean ("GEO") or a generalized linear model ("GLM"). Default is "GEO".

n_comp [numeric] Number indicating the number of principal components to compute. Cannot be more than n-1 samples. Default is 10.

remove_outliers [logical] A boolean to indicate if outliers should be excluded.

outliers_thresholds [list] List of thresholds to exclude outliers.

Details

Outliers definition (`remove_outliers = TRUE`):

- Binding Density (BD) < 0.1
- Binding Density (BD) > 2.25
- Field of View (FoV) < 75
- Positive Control Linearity (PCL) < 0.95
- Limit of Detection (LoD) < 2
- Positive normalisation factor (Positive_factor) < 0.25
- Positive normalisation factor (Positive_factor) > 4
- Housekeeping normalisation factor (house_factor) < 1/11
- Housekeeping normalisation factor (house_factor) > 11

Value

[list] A list containing parameters and data.

access [character] Value passed to `load_rcc()` in `id_colname`.

housekeeping_genes [character] Value passed to `load_rcc()` or `normalise()`.

housekeeping_predict [logical] Value passed to `load_rcc()`.

housekeeping_norm [logical] Value passed to `load_rcc()` or `normalise()`.

normalisation_method [character] Value passed to `load_rcc()` or `normalise()`.

remove_outliers [logical] Value passed to `normalise()`.

n_comp [numeric] Value passed to `load_rcc()`.

data_directory [character] Value passed to `load_rcc()`.

pc_sum [data.frame] A data.frame with `n_comp` rows and four columns: "Standard deviation", "Proportion of Variance", "Cumulative Proportion" and "PC".

nacho [data.frame] A data.frame with all columns from the sample sheet `ssheet_csv` and all computed columns, *i.e.*, quality-control metrics and counts, with one sample per row.

outliers_thresholds [list] A list of the quality-control thresholds used.

raw_counts [data.frame] Raw counts with probes as rows and samples as columns. With "CodeClass" (first column), the type of the probes and "Name" (second column), the Name of the probes.

normalised_counts [data.frame] Normalised counts with probes as rows and samples as columns. With "CodeClass" (first column), the type of the probes and "Name" (second column), the name of the probes.

Examples

```
data(GSE74821)
GSE74821_norm <- normalise(
  nacho_object = GSE74821,
  housekeeping_norm = TRUE,
  normalisation_method = "GEO",
  remove_outliers = TRUE
)

if (interactive()) {
  library(GEOquery)
  library(NACHO)

  # Import data from GEO
  gse <- GEOquery::getGEO(GEO = "GSE74821")
  targets <- Biobase::pData(Biobase::phenoData(gse[[1]]))
  GEOquery::getGEOSuppFiles(GEO = "GSE74821", baseDir = tempdir())
  utils::untar(
    tarfile = file.path(tempdir(), "GSE74821", "GSE74821_RAW.tar"),
    exdir = file.path(tempdir(), "GSE74821")
  )
  targets$IDFILE <- list.files(
    path = file.path(tempdir(), "GSE74821"),
    pattern = ".RCC.gz$"
```

```

)
targets[] <- lapply(X = targets, FUN = iconv, from = "latin1", to = "ASCII")
utils::write.csv(
  x = targets,
  file = file.path(tempdir(), "GSE74821", "Samplesheet.csv")
)

# Read RCC files and format
nacho <- load_rcc(
  data_directory = file.path(tempdir(), "GSE74821"),
  ssheet_csv = file.path(tempdir(), "GSE74821", "Samplesheet.csv"),
  id_colname = "IDFILE"
)

# (re)Normalise data by removing outliers
nacho_norm <- normalise(
  nacho_object = nacho,
  remove_outliers = TRUE
)

# (re)Normalise data with "GLM" method and removing outliers
nacho_norm <- normalise(
  nacho_object = nacho,
  normalisation_method = "GLM",
  remove_outliers = TRUE
)
}

```

print.nacho

Print method for "nacho" object

Description

This function allows to print text and figures from the results of a call to [load_rcc\(\)](#) or [normalise\(\)](#). It is intended to be used in a Rmarkdown chunk.

Usage

```

## S3 method for class 'nacho'
print(
  x,
  colour = "CartridgeID",
  size = 0.5,
  show_legend = FALSE,
  show_outliers = TRUE,
  outliers_factor = 1,
  outliers_labels = NULL,
  echo = FALSE,

```

```

    title_level = 1,
    xaringan = FALSE,
    ...
)

```

Arguments

x	[list] A list object of class "nacho" obtained from <code>load_rcc()</code> or <code>normalise()</code> .
colour	[character] Character string of the column in <code>ssheet_csv</code> or more generally in <code>nacho_object\$nacho</code> to be used as grouping colour.
size	[numeric] A numeric controlling point size (<code>ggplot2::geom_point()</code> or line size (<code>ggplot2::geom_line()</code>).
show_legend	[logical] Boolean to indicate whether the plot legends should be plotted (TRUE) or not (FALSE). Default is TRUE.
show_outliers	[logical] Boolean to indicate whether the outliers should be highlighted in red (TRUE) or not (FALSE). Default is TRUE.
outliers_factor	[numeric] Size factor for outliers compared to size. Default is 1.
outliers_labels	[character] Character to indicate which column in <code>nacho_object\$nacho</code> should be used to be printed as the labels for outliers or not. Default is NULL.
echo	[logical] A boolean to indicate whether text and plots should be printed. Mainly for use within a Rmarkdown chunk.
title_level	[numeric] A numeric to indicate the title level to start with, using markdown style, <i>i.e.</i> , the number of "#".
xaringan	[logical] A boolean to format output for Xaringan slides.
...	Other arguments (<i>Not used</i>).

Examples

```

data(GSE74821)
print(GSE74821)

```

render

Render a HTML report of a "nacho" object

Description

This function create a Rmarkdown script and render it as a HTML document. The HTML document is a quality-control report using all the metrics from `visualise()` based on recommendations from NanoString.

Usage

```
render(
  nacho_object,
  colour = "CartridgeID",
  output_file = "NACHO_QC.html",
  output_dir = ".",
  size = 1,
  show_legend = TRUE,
  show_outliers = TRUE,
  outliers_factor = 1,
  outliers_labels = NULL,
  clean = TRUE
)
```

Arguments

nacho_object	[list] A list object of class "nacho" obtained from load_rcc() or normalise() .
colour	[character] Character string of the column in <code>ssheet_csv</code> or more generally in <code>nacho_object\$nacho</code> to be used as grouping colour.
output_file	[character] The name of the output file.
output_dir	[character] The output directory for the rendered output_file. This allows for a choice of an alternate directory to which the output file should be written (the default output directory is the working directory, <i>i.e.</i> , <code>.</code>). If a path is provided with a filename in <code>output_file</code> the directory specified here will take precedence. Please note that any directory path provided will create any necessary directories if they do not exist.
size	[numeric] A numeric controlling point size (ggplot2::geom_point()) or line size (ggplot2::geom_line()).
show_legend	[logical] Boolean to indicate whether the plot legends should be plotted (TRUE) or not (FALSE). Default is TRUE.
show_outliers	[logical] Boolean to indicate whether the outliers should be highlighted in red (TRUE) or not (FALSE). Default is TRUE.
outliers_factor	[numeric] Size factor for outliers compared to size. Default is 1.
outliers_labels	[character] Character to indicate which column in <code>nacho_object\$nacho</code> should be used to be printed as the labels for outliers or not. Default is NULL.
clean	[logical] Boolean to indicate whether the Rmd and Rdata file used to produce the HTML report are removed from <code>output_dir</code> . Default is TRUE.

Examples

```
if (interactive()) {
  data(GSE74821)
  render(GSE74821)
}
```

visualise*Visualise quality-control metrics of a "nacho" object*

Description

This function allows to visualise results from `load_rcc()` or `normalise()` several quality-control metrics in an interactive shiny application, in which thresholds can be customised and exported.

Usage

```
visualise(nacho_object)
```

Arguments

`nacho_object` [\[list\]](#) A list object of class "nacho" obtained from `load_rcc()` or `normalise()`.

Examples

```
if (interactive()) {
  data(GSE74821)
  # Must be run in an interactive R session!
  visualise(GSE74821)
}

if (interactive()) {
  library(GEOquery)
  library(NACHO)

  # Import data from GEO
  gse <- GEOquery::getGEO(GEO = "GSE74821")
  targets <- Biobase::pData(Biobase::phenoData(gse[[1]]))
  GEOquery::getGEOSuppFiles(GEO = "GSE74821", baseDir = tempdir())
  utils::untar(
    tarfile = file.path(tempdir(), "GSE74821", "GSE74821_RAW.tar"),
    exdir = file.path(tempdir(), "GSE74821")
  )
  targets$IDFILE <- list.files(
    path = file.path(tempdir(), "GSE74821"),
    pattern = ".RCC.gz$"
  )
  targets[] <- lapply(X = targets, FUN = iconv, from = "latin1", to = "ASCII")
  utils::write.csv(
    x = targets,
    file = file.path(tempdir(), "GSE74821", "Samplesheet.csv")
  )

  # Read RCC files and format
  nacho <- load_rcc(
    data_directory = file.path(tempdir(), "GSE74821"),
    ssheet_csv = file.path(tempdir(), "GSE74821", "Samplesheet.csv"),
```

```
      id_colname = "IDFILE"
    )
    visualise(nacho)

    # (re)Normalise data by removing outliers
    nacho_norm <- normalise(
      nacho_object = nacho,
      remove_outliers = TRUE
    )
    visualise(nacho_norm)

    # (re)Normalise data with "GLM" method and removing outliers
    nacho_norm <- normalise(
      nacho_object = nacho,
      normalisation_method = "GLM",
      remove_outliers = TRUE
    )
    visualise(nacho_norm)
  }
}
```

Index

* datasets

GSE74821, [5](#)

autoplot.nacho, [2](#)

character, [3](#), [4](#), [6](#), [8](#), [9](#), [11](#), [12](#)

check_outliers, [4](#)

data.frame, [6](#), [9](#)

deploy, [4](#)

ggplot2::geom_line(), [3](#), [11](#), [12](#)

ggplot2::geom_point(), [3](#), [11](#), [12](#)

GSE74821, [5](#)

list, [3–6](#), [8](#), [9](#), [11–13](#)

load_rcc, [5](#)

load_rcc(), [3](#), [4](#), [6–13](#)

logical, [3](#), [4](#), [6](#), [8](#), [9](#), [11](#), [12](#)

normalise, [7](#)

normalise(), [3](#), [4](#), [8–13](#)

normalize (normalise), [7](#)

numeric, [3](#), [6](#), [8](#), [9](#), [11](#), [12](#)

print.nacho, [10](#)

render, [11](#)

render(), [2](#)

visualise, [13](#)

visualise(), [2](#), [11](#)

visualize (visualise), [13](#)