Package: qp (via r-universe)

November 3, 2024

Title A toolkit for analyzing protein quantification results

Version 0.1.0

Description What the package does (one paragraph).

License MIT + file LICENSE

Encoding UTF-8

Language en

Roxygen list(markdown = TRUE)

RoxygenNote 7.3.1

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

Config/testthat/edition 3

URL https://kaiaragaki.github.io/qp/, https://github.com/KaiAragaki/qp

Imports dplyr (>= 1.1.0), ggplot2, rlang, gplate, mop, tidyr, cli, gt, purrr

Remotes KaiAragaki/gplate, KaiAragaki/mop

Depends R (>= 2.10)

LazyData true

VignetteBuilder knitr

Repository https://kaiaragaki.r-universe.dev

RemoteUrl https://github.com/KaiAragaki/qp

RemoteRef HEAD

RemoteSha 98d820ab2cf2f302738cba9e18458814e949e60f

Contents

`absorbances` *Absorbances from a protein quantification*

Description

Absorbances from a BCA protein quantification in a `data.frame`

Usage

```
absorbances
```

Format

`absorbances`**:**

A `data.frame` with 96 rows and 5 columns:;

.row The row of the 96 well plate, where 1 refers to the top row.

.col The column of the 96 well plate, where 1 refers to the left column.

.abs The absorbance of the contents of the well at 562nm.

sample_type Denotes whether the sample is a standard or an unknown (sample).

index Denotes individual standards/samples, where each gets its own index.

`abs_to_col` *Convert an absorbance to a hexidecimal color*

Description

Takes an absorbance and converts it to a hexidecimal color. For the default qp_pal palette, this should provide a color that approximates real life color at the given absorbance.

Usage

```
abs_to_col(abs, pal)
```

Arguments

`abs`	Numeric. Absorbances.
`pal`	Character. A vector of hexidecimal colors.

Details

The absorbances have typical baseline absorbance (~ 0.07) removed, and then an index is calculated with a logistic curve of maximum 100 and a center of 0.15.

Value

Character. Hexidecimal colors corresponding to absorbances.

`dilute` *Calculate dilution from known concentrations*

Description

Calculate dilution from known concentrations

Usage

```
dilute(c1, c2 = min(c1), v2, round_for_pipettes = TRUE)
```

Arguments

`c1`	Numeric. Initial concentration of sample.
`c2`	Numeric. Target concentration of sample.
`v2`	Numeric. Target final volume of sample. If `round_for_pipettes = TRUE`, assumes volume is uL.
`round_for_pipettes`	Logical. If TRUE, rounds values to the accuracy of standard pipettes using `make_pipette_vol`.

Value

a data.frame, with `sample_to_add` as the volume of sample to add, and `add_to` as the volume to dilute the sample into.

Examples

```
dilute(203, 70, 10)
dilute(203, 70, 10, round_for_pipettes = FALSE)
# Vectorized:
dilute(c(8, 10, 12), c(4, 5, 6), c(7, 8, 9))
```

`make_pipette_vol` *Round volume to be pipette-compatible*

Description

Round volume to be pipette-compatible

Usage

```
make_pipette_vol(x)
```

Arguments

`x`	Numeric. Volume to be rounded

Value

Numeric. Rounded volume.

Examples

```
make_pipette_vol(104.13398)
make_pipette_vol(15.3331)
make_pipette_vol(9.9211)
# Vectorized:
make_pipette_vol(c(104.13398, 15.3331, 9.9211, NA, -100.1))
```

qp — *Quantify protein concentration from a MicroBCA assay*

Description

Quantify protein concentration from a MicroBCA assay

Usage

```
qp(
  x,
  replicate_orientation = c("h", "v"),
  sample_names = NULL,
  remove_empty = TRUE,
  ignore_outliers = c("all", "samples", "standards", "none"),
  standard_scale = c(0, 2^((2:7) - 5)),
  n_replicates = 3,
  wavelength = 562
)
```

Arguments

`x`	A `spectramax`, gp, or `data.frame` object, or path to SPECTRAmax .xls(x)/.txt file.
`replicate_orientation`	Either 'h' or 'v' - see Details.
`sample_names`	Optional character vector of sample names.
`remove_empty`	Should wells that have less absorbance than the lowest standard be dropped?
`ignore_outliers`	Character. From which group - samples or standards - should outliers be detected and removed?
`standard_scale`	Numeric. Known concentrations of standards, in the order they appear.
`n_replicates`	Numeric. The number of techinical replicates.
`wavelength`	Numeric. The wavelength absorbance was captured.

Details

If `x` is a `spectramax`, the standards must start in the upper left corner in the order dictated by `standard_scale`. Whether this is from from left to right or top to bottom can be specified in `replicate_orientation`. Note that `replicate_orientation` specified the direction that REPLICATES lie, NOT the direction the samples flow (which will be perpendicular to the replicates).

Note: `replicate_orientation`, `n_replicates`, and `wavelength` will be silently ignored if `x` is not a `spectramax` or path to a `spectramax`

Value

a `tibble`

Examples

```
data <- system.file("extdata", "absorbances.txt", package = "qp")
qp(data, replicate_orientation = "h")
```

`qp_add_names` *Add sample names*

Description

Add sample names

Usage

```
qp_add_names(x, ...)

## S3 method for class 'list'
qp_add_names(x, sample_names = NULL, ...)

## S3 method for class 'data.frame'
qp_add_names(x, sample_names = NULL, ...)
```

Arguments

`x`	A `data.frame` (or a list containing one) that contains columns `index` (which denotes sample number) and `sample_type`, which should be either "unknown" or "standard".
`...`	Unused
`sample_names`	Optional character vector. If NULL, uses sample index. In a standard workflow, the index is the order the sample appears in the plate

Examples

```
df <- expand.grid(
  index = c(1, 1, 2, 2, 2, 3),
  sample_type = c("standard", "unknown")
)

df

# You don't get to name standards:
qp_add_names(df, c("a", "b", "c"))

# If there aren't enough names, will use index
```

```
qp_add_names(df, c("a", "b"))

# No names provided will use index by default
qp_add_names(df)
```

---

qp_add_std_conc *Add known concentrations of protein to standard samples*

Description

Add known concentrations of protein to standard samples

Usage

```
qp_add_std_conc(x, standard_scale = c(0, 2^((2:7) - 5)), ...)

## S3 method for class 'data.frame'
qp_add_std_conc(x, standard_scale = c(0, 2^((2:7) - 5)), ...)

## S3 method for class 'list'
qp_add_std_conc(x, standard_scale = c(0, 2^((2:7) - 5)), ...)
```

Arguments

`x`	A `data.frame` containing a `sample_type` and `index` columns. See details.
`standard_scale`	A numeric vector giving the concentrations of the standards. The units are arbitrary, but will determine the units of the output concentrations.
`...`	Unused

Details

Input is expected to have two columns:

- `sample_type`: A character vector denoting which samples are standards with "standard". All other values will be considered unknowns.
- `index`: A numeric column denoting the sample number. Index 1 will correspond to the first item in `standard_scale`, 2 will be the second, etc.

Value

Same type as x, with a `.conc` column

Examples

```
abs <- expand.grid(
  sample_type = c("standard", "unknown"),
  index = 1:7
)

abs

qp_add_std_conc(abs)

# Can add custom scale - doesn't have to be 'in order' or unique:
qp_add_std_conc(abs, c(1, 4, 2, 2, 3, 0.125, 7))

# Will warn - more values in `standard_scale` than standard indices
# Will drop extra
qp_add_std_conc(abs, 1:8)

# Will error - fewer values in `standard_scale` than standard indices
if (FALSE) {
  qp_add_std_conc(abs, 1:6)
}
```

`qp_calc_abs_mean` *Calculate absorbance means with optional outlier removal*

Description

Calculate absorbance means with optional outlier removal

Usage

```
qp_calc_abs_mean(x, ignore_outliers = c("all", "standards", "samples", "none"))

## S3 method for class 'data.frame'
qp_calc_abs_mean(x, ignore_outliers = c("all", "standards", "samples", "none"))

## S3 method for class 'list'
qp_calc_abs_mean(x, ignore_outliers = c("all", "standards", "samples", "none"))
```

Arguments

`x`	A `data.frame` or `list` containing a `data.frame` named qp. See details.
`ignore_outliers`	Which sample types should have outliers ignored from their mean calculations? If `.is_outlier` column is supplied, this argument is ignored.

Details

Input `data.frame` must contain the following columns:

- `sample_type`. Character. Must contain values either "standard" or "unknown"
- `index`. Numeric. Denotes sample number.
- `.abs`. Numeric. Contains absorbance values.
- If a boolean `.is_outlier` is supplied, that will be used instead.

Value

The input `tibble` with an `.is_outlier` column and a `.mean` column

Examples

```
 library(dplyr)

abs <- expand.grid(
  sample_type = c("standard", "unknown"),
  index = 1:7,
  rep = 1:3
) |>
  dplyr::arrange(sample_type, index, rep)

abs$.abs <- abs(rnorm(nrow(abs), mean = abs$index))

# Selecting different subsets for outlier removal
qp_calc_abs_mean(abs, "none")

qp_calc_abs_mean(abs, "standards")

qp_calc_abs_mean(abs, "samples")

qp_calc_abs_mean(abs, "all")

# If an `.is_outlier` column is provided, that will be used instead:

abs$.is_outlier <- rep(c(TRUE, FALSE), length.out = nrow(abs))

qp_calc_abs_mean(abs)
```

`qp_calc_conc` *Predict concentrations from standards fit*

Description

Predict concentrations from standards fit

Usage

```
qp_calc_conc(x, ignore_outliers = TRUE, group_cols = c("sample_type", "index"))
```

Arguments

`x`	A list. See details.
`ignore_outliers`	Boolean. Should outliers be considered when calculating the mean? See details.
`group_cols`	Character vector. Columns to group by before taking the mean.

Details

The supplied list should contain two items - `fit`, generated by `qp_fit`, and `qp`, a data.frame. `qp` should contain the following:

- Columns used in `fit`. Usually, this is `.log2_abs`
- Any columns in `group_cols`
- If `ignore_outliers = TRUE`, `.is_outlier` will be used if supplied, or created if not.

Value

Returns a `list` with the input fit and data.frame, with additional columns:

- `.pred`: The predicted value from the provided model
- `.pred_conc`: `.pred`, transformed by `conc_transform`
- `.pred_conc_mean`: The mean of `.pred_conc`, sans samples where column `.is_outlier == TRUE`

Examples

```
data <- system.file("extdata", "absorbances.txt", package = "qp")
calculated <- qp(data, replicate_orientation = "h")

# Making a minimal object:
calculated$qp <- calculated$qp |>
  dplyr::select(
    .log2_abs, sample_type, index, .is_outlier
  )

calculated

qp_calc_conc(calculated)
```

qp_dilute *Calculate dilutions from predicted concentrations*

Description

Calculate dilutions from predicted concentrations

Usage

```
qp_dilute(x, ...)

## S3 method for class 'data.frame'
qp_dilute(
  x,
  target_conc = NULL,
  target_vol = 15,
  remove_standards = FALSE,
  pipette_vol_compat = TRUE,
  ...
)

## S3 method for class 'list'
qp_dilute(
  x,
  target_conc = NULL,
  target_vol = 15,
  remove_standards = FALSE,
  ...
)
```

Arguments

`x`	A `data.frame` or `list` containing a `data.frame` named `qp` with a column named `.pred_conc` or `.pred_conc_mean`. If both, will favor `.pred_conc_mean`.
`...`	Unused
`target_conc`	Numeric vector. Target concentration in (mg/mL) protein. If length == 1, recycled.
`target_vol`	Target volume in uL. If length == 1, recycled.
`remove_standards`	Boolean. Should standards be removed from results?
`pipette_vol_compat`	Boolean. Shold returned numbers be rounded to the typically precision of a pipette?

Value

Same as input, with the volumes of lysate and volumes of diluent to add.

Examples

```
df <- data.frame(.pred_conc = 1)
qp_dilute(df, target_conc = 0.5, target_vol = 30)


# Many sample and target concentrations
df2 <- data.frame(.pred_conc = 1:3)
qp_dilute(df2, target_conc = c(0.1, 0.4, 0.8), target_vol = 30)

# Takes a list, so long as it has a data.frame named qp as one of the items:
ls <- list(qp = data.frame(.pred_conc = 3))
qp_dilute(ls, target_conc = 0.5, target_vol = 30)
```

`qp_fit` *Fit an lm using standards absorbances*

Description

Fit an lm using standards absorbances

Usage

```
qp_fit(x)

## S3 method for class 'data.frame'
qp_fit(x)

## S3 method for class 'list'
qp_fit(x)
```

Arguments

x	A `data.frame` or `list` containing a `data.frame` under the name qp. See details.

Details

The supplied `data.frame` must have the following columns:

- `sample_type`. Character. If not 'standard', assumed to be a sample
- `.is_outlier`. Boolean. If TRUE, assumed to be outlier and removed from fitting. If FALSE or NA, used for fitting. If unsupplied, will create one with all values set to NA.
- `.conc`. Numeric. Known concentration of standard.
- `.log2_abs`. Numeric. The log2 of the absorbances

Value

A list containing:

- `fit`, an `lm` object fit with the formula `.log2_conc` ~ `.log2_abs`, fit using non-outlier standards
- qp, the input data

Examples

```
absorbances |>
  qp_add_std_conc() |>
  qp_fit()
```

`qp_mark_outliers` *Mark absorbance outliers*

Description

Mark absorbance outliers

Usage

```
qp_mark_outliers(x, ignore_outliers = c("all", "standards", "samples", "none"))

## S3 method for class 'data.frame'
qp_mark_outliers(x, ignore_outliers = c("all", "standards", "samples", "none"))

## S3 method for class 'list'
qp_mark_outliers(x, ignore_outliers = c("all", "standards", "samples", "none"))
```

Arguments

`x`	A `data.frame` or `list` containing a `data.frame` named qp. See details.
`ignore_outliers`	Which sample types should have outliers marked?

Details

Input `data.frame` must contain the following columns:

- `sample_type`. Character. Must contain values either "standard" or "unknown"
- `index`. Numeric. Denotes sample number.
- `.abs`. Numeric. Contains absorbance values.

Value

The input `tibble` with an `.is_outlier` column

Examples

```
df <- data.frame(
  sample_type = rep(c("standard", "unknown"), each = 3),
  index = c(1, 1, 1, 2, 2, 2),
  .abs = c(1, 1, 1, 1, 1, 2)
)

qp_mark_outliers(df, ignore_outliers = "all")
qp_mark_outliers(df, ignore_outliers = "standards")
qp_mark_outliers(df, ignore_outliers = "samples")
qp_mark_outliers(df, ignore_outliers = "none")
```

`qp_pal` *The default color palette for qp*

Description

It attempts to match the real life colors of a protein quantification experiment, in combination with `abs_to_col`

Usage

```
qp_pal
```

Format

An object of class `character` of length 100.

`qp_plot_plate` *View the absorbances of an analyzed* `qp` *as they were on the plate*

Description

View the absorbances of an analyzed `qp` as they were on the plate

Usage

```
qp_plot_plate(x, size = 15)
```

Arguments

`x`	A `data.frame` with `.row`, `.col`, and `.abs` columns
`size`	The size of the points used to illustrate the wells. Passed to geom_point.

Value

a `ggplot`

Examples

```
qp_plot_plate(absorbances)
```

`qp_plot_standards` *View an absorbance/concentration plot*

Description

View an absorbance/concentration plot

Usage

```
qp_plot_standards(x)
```

Arguments

`x`	The output of `qp` or `qp_calc_conc`

Value

a `ggplot`

Examples

```
absorbances |>
  qp() |>
  qp_plot_standards()
```

`qp_remove_empty` *Remove empty wells from data*

Description

Remove empty wells from data

Usage

```
qp_remove_empty(x)

## S3 method for class 'data.frame'
qp_remove_empty(x)

## S3 method for class 'list'
qp_remove_empty(x)
```

Arguments

`x`	A `data.frame` or `list` containing a `data.frame` named qp containing columns `.pred_conc` and `sample_type`. See details.

Details

This function keeps any columns with positive `.pred_conc` or `sample_type == "standard"`

Value

Same as input

Examples

```
df <- expand.grid(
  .pred_conc = 0:1,
  sample_type = c("standard", "unknown")
)

df

qp_remove_empty(df)
```

`qp_report` *Create a report for a protein quantificaiton experiment*

Description

Create a report for a protein quantificaiton experiment

Usage

```
qp_report(qp, output_file, other = list())
```

Arguments

`qp`	Likely the output from qp AND `qp_dilute`.
`output_file`	Character. The path of the file to export, including `.html`
`other`	Generally used for Shiny application. Assumes a named list of key-values that will be used to document report parameters.

Examples

```
## Not run:
absorbances |>
  qp() |>
  qp_dilute() |>
  qp_report(
    "~/my_report.html",
    other = list(key = "value") # Essentially metadata
  )

## End(Not run)
```

`qp_summarize` *Summarize output from qp pipeline*

Description

Summarize output from qp pipeline

Usage

```
qp_summarize(x)

## S3 method for class 'data.frame'
qp_summarize(x)

## S3 method for class 'list'
qp_summarize(x)
```

Arguments

x	A `data.frame` or a `list` containing a `data.frame` named qp

Value

A `tibble` with the sample `name`, `sample_type`, and the mean of its predicted concentration (`.pred_conc_mean`)

`qp_tidy` *Read in and wrangle protein quantification data*

Description

Read in and wrangle protein quantification data

Usage

```
qp_tidy(x, ...)

## S3 method for class 'character'
qp_tidy(x, ...)

## S3 method for class 'spectramax'
qp_tidy(
  x,
  replicate_orientation = c("h", "v"),
  n_standards = 7,
  n_replicates = 3,
  wavelength = 562,
  ...
)

## S3 method for class 'gp'
qp_tidy(x, ...)

## Default S3 method:
qp_tidy(x, ...)
```

Arguments

`x`	A gp, `data.frame/tibble`, `spectramax`, or character path to a raw SPECTRAmax .xls(x)/.txt
`...`	Arguments passed to relevant methods.
`replicate_orientation`	Character. Specified the direction the *replicates* lie, not the direction the samples flow (which will be perpendicular to `replicate_orientation`).
`n_standards`	Numeric. The number of different concentrations of standards. Does not include replicates.
`n_replicates`	Numeric. The number of replicates per sample.
`wavelength`	Numeric. For SPECTRAmax files and objects, the wavelength measured. Otherwise, ignored.

Details

qp assumes that if you read in data not in a `spectramax` file or object, you probably have a custom workflow in mind - therefore, tidying will be minimal and mostly focused on checking for validity.

Value

a `data.frame`

Examples

```
data <- system.file("extdata", "absorbances.txt", package = "qp")
qp_tidy(data)
```

Index