Package ‘rtpcr’

February 24, 2026

Type Package

Title qPCR Data Analysis

Version 2.1.5

Description Tools for qPCR data analysis using Delta Ct and Delta Delta Ct methods, including t-test, Wilcoxon-test, ANOVA models, and publication-ready visualizations. The package supports multiple target, and multiple reference genes, and uses a calculation framework adopted from Ganger et al. (2017) <doi:10.1186/s12859-017-1949-5> and Taylor et al. (2019) <doi:10.1016/j.tibtech.2018.12.002>, covering both the Livak and Pfaffl methods.

URL https://mirzaghaderi.github.io/rtpcr/,

https://github.com/mirzaghaderi/rtpcr

License GPL-3

Imports multcomp, ggplot2, lme4, lmerTest, purrr, reshape2, tidyr, dplyr, grid, emmeans, lifecycle

Encoding UTF-8

RoxygenNote 7.3.3

NeedsCompilation no

Author Ghader Mirzaghaderi [aut, cre, cph]

Depends R (>= 3.5.0)

Suggests knitr, rmarkdown, multcompView

VignetteBuilder knitr

LazyData true

Maintainer Ghader Mirzaghaderi <mirzaghaderi@gmail.com>

Repository CRAN

Date/Publication 2026-02-24 09:30:08 UTC

Contents

ANOVA_DCt 2
ANOVA_DDCt 4

compute_wDCt 8
data_2factorBlock3ref 9
efficiency 10
long_to_wide 11
Means_DDCt 12
meanTech 13
multiplot 15
plotFactor 17
TTEST_DDCt 19
WILCOX_DDCt 22

Index **24**

`ANOVA_DCt` *Delta Ct ANOVA analysis with optional model specification*

Description

Performs Delta Ct (dCt) analysis of the data from a 1-, 2-, or 3-factor experiment with support for both fixed effects and mixed effects models. Per-gene statistical grouping is performed for all treatment combinations.

Usage

```
ANOVA_DCt(
  x,
  numOfFactors,
  numberOfrefGenes,
  block = NULL,
  alpha = 0.05,
  p.adj = "none",
  analyseAllTarget = TRUE,
  model = NULL,
  modelBased_se = TRUE,
  set_missing_target_Ct_to_40 = FALSE
)
```

Arguments

`x` The input data frame containing experimental design columns, target gene E/Ct column pairs, and reference gene E/Ct column pairs. Reference gene columns must be located at the end of the data frame. See "Input data structure" in vignettes for details about data structure.

`numOfFactors` Integer. Number of experimental factor columns (excluding `rep` and optional `block`).

`numberOfrefGenes` Integer. Number of reference genes. Each reference gene must be represented by two columns (E and Ct).

`block` Character. Block column name or `NULL`. When a qPCR experiment is done in multiple qPCR plates, variation resulting from the plates may interfere with the actual amount of gene expression. One solution is to conduct each plate as a randomized block so that at least one replicate of each treatment and control is present on a plate. Block effect is usually considered as random and its interaction with any main effect is not considered. Note: This parameter is ignored if `model` is provided.

`alpha` Statistical level for comparisons (default: 0.05).

`p.adj` Method for p-value adjustment. See `p.adjust`.

`analyseAllTarget` Logical or character. If `TRUE` (default), all detected target genes are analysed. Alternatively, a character vector specifying the names (names of their Efficiency columns) of target genes to be analysed.

`model` Optional model formula. If provided, this overrides the automatic formula (CRD or RCBD based on `block` and `numOfFactors`). The formula uses `wDCt` as the response variable. For mixed models, random effects can be defined using `lmer` syntax (e.g., `"wDCt ~ Treatment + (1|Block)"`). When using `model`, the `block` and `numOfFactors` arguments are ignored for model specification, but still used for data structure identification.

`modelBased_se` Logical. If `TRUE` (default), standard errors are calculated from model-based residuals. If `FALSE`, standard errors are calculated directly from the observed `wDCt` values within each treatment group according to the selected `se.type`. For single factor data, both methods are the same. It is recommended to use `modelBased_se = TRUE` (default).

`set_missing_target_Ct_to_40` If `TRUE`, missing target gene Ct values become 40; if `FALSE` (default), they become NA.

Details

The function performs ANOVA analysis on weighted delta Ct (wDCt) values and returns variance components along with an expression table containing:

- `gene`: Name of target genes
- Factor columns: Experimental design factors
- `dCt`: Mean weighted delta Ct for each treatment combination
- `RE`: Relative expression = 2^-dCt
- `log2FC`: log2 of relative expression
- `LCL`: 95% lower confidence level
- `UCL`: 95% upper confidence level
- `se`: Standard error of the mean calculated from wDCt values
- `Lower.se.RE`: Lower limit error bar for RE (2^(log2(RE) - se))
- `Upper.se.RE`: Upper limit error bar for RE (2^(log2(RE) + se))
- `Lower.se.log2FC`: Lower limit error bar for log2 RE
- `Upper.se.log2FC`: Upper limit error bar for log2 RE
- `sig`: Per-gene significance grouping letters

Value

An object containing expression tables, lm/lmer models, ANOVA tables, residuals, and raw data for each gene:

`relativeExpression` dCt expression table for all treatment combinations along with per-gene statistical grouping

`perGene` Nested list containing detailed results for each target gene:

- `ANOVA_table`: Full factorial ANOVA table
- `lm`: lm/lmer model for factorial design
- `Final_data`: Processed data with wDCt values
- `resid(object$perGene$gene_name$lm)`: Residuals

Examples

```
# Default usage with fixed effects
result <- ANOVA_DCt(data_2factorBlock3ref, numOfFactors = 2, numberOfrefGenes = 3,
                    block = "block")

# Mixed model with random block effect
result_mixed <- ANOVA_DCt(data_2factorBlock3ref, numOfFactors = 2, numberOfrefGenes = 3,
                          block = "block")

# Custom mixed model formula with nested random effects
result_custom <- ANOVA_DCt(data_repeated_measure_2, numOfFactors = 2, numberOfrefGenes = 1,
                             block = NULL,
                             model = wDCt ~ treatment * time + (1 | id))
```

ANOVA_DDCt — *Delta Delta Ct ANOVA analysis with optional model specification*

Description

Apply Delta Delta Ct (ddCt) analysis to each target gene and performs per-gene statistical analysis.

Usage

```
ANOVA_DDCt(
  x,
  numOfFactors,
  numberOfrefGenes,
  specs,
  block,
  calibratorLevel = NULL,
  p.adj = "none",
  analyseAllTarget = TRUE,
```

```
  model = NULL,
  set_missing_target_Ct_to_40 = FALSE,
  se.type = c("single.group", "paired.group", "two.group"),
  modelBased_se = TRUE,
  ...
)
```

Arguments

`x` The input data frame containing experimental design columns, replicates (integer), target gene E/Ct column pairs, and reference gene E/Ct column pairs. Reference gene columns must be located at the right end of the data frame. See "Input data structure" in vignettes for details about data structure.

`numOfFactors` Integer. Number of experimental factor columns (excluding `rep` and optional `block`).

`numberOfrefGenes` Integer. Number of reference genes.

`specs` Example: "A", "A|B" or A|B*C" if A, B and C are name of factor columns in the input data The first name (here A) is the factor for which the relative expression analysis is applied.

`block` Character. Block column name or `NULL`. When a qPCR experiment is done in multiple qPCR plates, variation resulting from the plates may interfere with the actual amount of gene expression. One solution is to conduct each plate as a randomized block so that at least one replicate of each treatment and control is present on a plate. Block effect is usually considered as random and its interaction with any main effect is not considered.

`calibratorLevel` NULL or one of the levels of the first selected factor in specs argument. If NULL the first level of that factor is used as calibrator. Optional character vector specifying the order of levels for the main factor. If `NULL`, the first observed level is used as the calibrator. If provided, the first element of the vector is used as the calibrator level.

`p.adj` Method for p-value adjustment. See `p.adjust`.

`analyseAllTarget` Logical or character. If `TRUE` (default), all target genes are analysed. Alternatively, a character vector specifying the names (names of their Efficiency columns) of target genes to be analysed.

`model` Optional model formula. If provided, this overrides the automatic formula (uni - or multi-factorial CRD or RCBD based on `block` and `numOfFactors`). The formula uses `wDCt` as the response variable. For mixed models, random effects can be defined using `lmer` syntax (e.g., `"wDCt ~ Treatment + (1 | id)"`). When using `model`, the `block` and `numOfFactors` arguments are ignored for model specification, but still used for data structure identification.

for fixed effects only, the `"lm"` (ordinary least squares) is used. `"lmer"` is used for mixed effects models (requires the `lmerTest` package). If a custom formula is provided with random effects, the function will use `lmerTest::lmer()`; otherwise it will use `stats::lm()`. Note that `emmeans` supports both model types and will use appropriate degrees of freedom methods (Satterthwaite by default).

`set_missing_target_Ct_to_40`
: If `TRUE`, missing target gene Ct values become 40; if `FALSE` (default), they become NA.

`se.type`
: Character string specifying how standard error is calculated. One of `"paired.group"`, `"two.group"`, or `"single.group"`. `"paired.group"` computes SE from paired differences (used when a random `id` effect is present), `"two.group"` uses the unpaired two-group t-test standard error against the reference level, and `"single.group"` computes SE within each level using a one-group t-test.

`modelBased_se`
: Logical. If `TRUE` (default), standard errors are calculated from model-based residuals. If `FALSE`, standard errors are calculated directly from the observed `wDCt` values within each treatment group according to the selected `se.type`. For single factor data, both methods are the same. It is recommended to use `modelBased_se = TRUE` (default).

`...`
: Additional arguments. Included for backward compatibility with deprecated `mainFactor.column`.

Details

ddCt analysis of variance (ANOVA) is performed for the `mainFactor.column` based on a full model factorial experiment by default. However, if `ANCOVA_DDCt` function is used, analysis of covariance is performed for the levels of the `mainFactor.column` and the other factors are treated as covariates. if the interaction between the main factor and the covariate is significant, ANCOVA is not appropriate.

All the functions for relative expression analysis (including `TTEST_DDCt()`, `WILCOX_DDCt()`, `ANOVA_DDCt()`, and `ANOVA_DCt()`) return the relative expression table which include fold change and corresponding statistics. The output of `ANOVA_DDCt()`, and `ANOVA_DCt()` also include lm models, residuals, raw data and ANOVA table for each gene.

The expression table returned by `TTEST_DDCt()`, `WILCOX_DDCt()`, and `ANOVA_DDCt()` functions include these columns: gene (name of target genes), contrast (calibrator level and contrasts for which the relative expression is computed), ddCt (mean of weighted delta delta Ct values), RE (relative expression or fold change = 2^-ddCt), log2FC (log(2) of relative expression or fold change), pvalue, sig (per-gene significance), LCL (95% lower confidence level), UCL (95% upper confidence level), se (standard error of mean calculated from the weighted delta Ct values of each of the main factor levels), Lower.se.RE (The lower limit error bar for RE which is 2^(log2(RE) - se)), Upper.se.RE (The upper limit error bar for RE which is 2^(log2(RE) + se)), Lower.se.log2FC (The lower limit error bar for log2 RE), and Upper.se.log2FC (The upper limit error bar for log2 RE)

Value

An object containing expression table, lm model, residuals, raw data and ANOVA table for each gene:

ddCt expression table along with per-gene statistical comparison outputs `object$relativeExpression`

ANOVA table `object$perGene$gene_name$ANOVA_table`

lm ANOVA `object$perGene$gene_name$lm`

lm_formula `object$perGene$gene_name$lm_formula`

Residuals `resid(object$perGene$gene_name$lm)`

References

LivakKJ, Schmittgen TD (2001). Analysis of Relative Gene Expression Data Using Real-Time Quantitative PCR and the Double Delta CT Method. *Methods*, 25(4), 402–408. doi:10.1006/meth.2001.1262

Ganger MT, Dietz GD, and Ewing SJ (2017). A common base method for analysis of qPCR data and the application of simple blocking in qPCR experiments. *BMC Bioinformatics*, 18, 1–11.

Taylor SC, Nadeau K, Abbasi M, Lachance C, Nguyen M, Fenrich, J. (2019). The ultimate qPCR experiment: producing publication quality, reproducible data the first time. *Trends in Biotechnology*, 37, 761-774.

Yuan JS, Reed A, Chen F, Stewart N (2006). Statistical Analysis of Real-Time PCR Data. *BMC Bioinformatics*, 7, 85.

Examples

```
data1 <- read.csv(system.file("extdata", "data_2factorBlock3ref.csv", package = "rtpcr"))
ANOVA_DDCt(data1,
           numOfFactors = 2,
           numberOfrefGenes = 3,
           block = "block",
           specs = "Concentration",
           p.adj = "none")

data2 <- read.csv(system.file("extdata", "data_1factor_one_ref.csv", package = "rtpcr"))
ANOVA_DDCt(data2,
           numOfFactors = 1,
           numberOfrefGenes = 1,
           block = NULL,
           specs = "Condition",
           p.adj = "none",
           se.type = "single.group")

# Repeated measure analysis
a <- ANOVA_DDCt(data_repeated_measure_1,
           numOfFactors = 1,
           numberOfrefGenes = 1,
           block = NULL,
           specs = "time",
           p.adj = "none", model = wDCt ~ time + (1 | id))

a$perGene$Target$ANOVA_table


# Repeated measure analysis: split-plot in time
a <- ANOVA_DDCt(data_repeated_measure_2,
           numOfFactors = 2, numberOfrefGenes = 1,
           specs = "time", block = NULL,
           model = wDCt ~ treatment * time + (1 | id))
```

compute_wDCt *Cleaning data and weighted delta Ct (wDCt) calculation*

Description

The compute_wDCt function cleans the data and computes wDCt. This function is automatically applied to the expression analysis functions like ANOVA_DDCt, TTEST_DDCt, etc. So it should not be applied in advance of expression analysis functions.

Usage

```
compute_wDCt(
  x,
  numOfFactors,
  numberOfrefGenes,
  block,
  set_missing_target_Ct_to_40 = FALSE
)
```

Arguments

x	A data frame containing experimental design columns, replicates (integer), target gene E/Ct column pairs, and reference gene E/Ct column pairs. Reference gene columns must be located at the end of the data frame.
numOfFactors	Integer. Number of experimental factor columns (excluding rep and optional block).
numberOfrefGenes	Integer. Number of reference genes.
block	Character or NULL. Name of the blocking factor column. When a qPCR experiment is done in multiple qPCR plates, each plate is considered as a random block so that at least one replicate of each treatment and control is present on a plate.
set_missing_target_Ct_to_40	If TRUE, missing target gene Ct values become 40; if FALSE (default), they become NA.

Details

The compute_wDCt function computes weighted delta Ct (wDCt) for the input data. Missing data can be denoted by NA in the input data frame. Values such as '0' and 'undetermined' (for any E and Ct) are automatically converted to NA. For target genes, NA for E or Ct measurements cause returning NA for the corresponding delta Ct for that replicate (row). If there are more than one reference gene, NA in the place of the E or the Ct value cause skipping that gene and remaining references are geometrically averaged. The compute_wDCt function is automatically applied to the expression analysis functions.

Value

The original data frame along with the weighted delta Ct column.

Examples

```
data <- read.csv(system.file("extdata", "data_2factorBlock3ref.csv", package = "rtpcr"))
data
compute_wDCt(x = data,
             numOfFactors = 2,
             numberOfrefGenes = 3,
             block = "block")
```

---

data_2factorBlock3ref *Sample data in (two factor with blocking factor and 3 reference genes)*

Description

A sample qPCR data set with blocking factor and 3 reference genes. Each line belongs to a separate individual (non-repeated measure experiment).

Usage

```
data_2factorBlock3ref
```

Format

A data frame with 18 observations and 8 variables:

Type First experimental factor

Concentration Second experimental factor

block blocking factor

Rep Biological replicates

PO Mean amplification efficiency of PO gene

Ct_PO Ct values of PO gene. Each is the mean of technical replicates

NLM Mean amplification efficiency of NLM gene

Ct_NLM Ct values of NLM gene. Each is the mean of technical replicates

ref1 Mean amplification efficiency of ref1 gene

Ct_ref1 Ct values of ref1 gene. Each is the mean of technical replicates

ref2 Mean amplification efficiency of ref2 gene

Ct_ref2 Ct values of ref2 gene. Each is the mean of technical replicates

ref3 Mean amplification efficiency of ref3 gene

Ct_ref3 Ct values of GAPDH gene. Each is the mean of technical replicates

Source

Not applicable

`efficiency` *Amplification efficiency statistics and standard curves*

Description

The `efficiency` function calculates amplification efficiency (E) and related statistics, including slope and coefficient of determination (R^2), and generates standard curves for qPCR assays.

Usage

```
efficiency(df, base_size = 12, legend_position = c(0.2, 0.2), ...)
```

Arguments

`df`	A data frame containing dilution series and corresponding Ct values. The first column should represent dilution levels, and the remaining columns should contain Ct values for different genes.
`base_size`	font size
`legend_position`	legend position
`...`	Additional ggplot2 layer arguments

Details

Amplification efficiency is estimated from standard curves generated by regressing Ct values against the logarithm of template dilution. For each gene, the function reports the slope of the standard curve, amplification efficiency (E), and R^2 as a measure of goodness of fit. The function also provides graphical visualization of the standard curves.

Value

A list with the following components:

efficiency A data frame containing slope, amplification efficiency (E), and R^2 statistics for each gene.

Slope_compare A table comparing slopes between genes.

plot A `ggplot2` object showing standard curves for all genes.

Author(s)

Ghader Mirzaghaderi

Examples

```
# Load example efficiency data
data <- read.csv(system.file("extdata", "data_efficiency1.csv", package = "rtpcr"))

# Calculate amplification efficiency and generate standard curves
efficiency(data)


ef <- read.csv(system.file("extdata", "data_efficiency_Yuan2006PMCBioinf.csv", package = "rtpcr"))
efficiency(ef)
```

---

long_to_wide — *Converts a 4-column qPCR long data format to wide format*

Description

Converts a 4-column (Condition, gene, Efficiency, Ct) qPCR long data format to wide format

Usage

```
long_to_wide(x)
```

Arguments

x	a 4-column (Condition, gene, Efficiency, Ct) qPCR long data

Details

Converts a 4-column (Condition, gene, Efficiency, Ct) qPCR long data format to wide format

Value

A wide qPCR data frame

Author(s)

Ghader Mirzaghaderi

Examples

```
df <- read.table(header = TRUE, text = "
Condition Gene E Ct
control C2H2-26 1.8 31.26
control C2H2-26 1.8 31.01
control C2H2-26 1.8 30.97
treatment C2H2-26 1.8 32.65
treatment C2H2-26 1.8 32.03
treatment C2H2-26 1.8 32.4
```

```
control C2H2-01 1.75 31.06
control C2H2-01 1.75 30.41
control C2H2-01 1.75 30.97
treatment C2H2-01 1.75 28.85
treatment C2H2-01 1.75 28.93
treatment C2H2-01 1.75 28.9
control C2H2-12 2 28.5
control C2H2-12 2 28.4
control C2H2-12 2 28.8
treatment C2H2-12 2 27.9
treatment C2H2-12 2 28
treatment C2H2-12 2 27.9
control ref 1.9 28.87
control ref 1.9 28.42
control ref 1.9 28.53
treatment ref 1.9 28.31
treatment ref 1.9 29.14
treatment ref 1.9 28.63")

long_to_wide(df)
```

`Means_DDCt` *Delta Delta Ct pairwise comparisons using a fitted model*

Description

Performs relative expression (fold change) analysis based on the Delta Delta Ct (ddCt) methods using a fitted model object produced by `ANOVA_DCt()` or `ANOVA_DDCt()`.

Usage

```
Means_DDCt(model, specs, p.adj = "none")
```

Arguments

`model`	A fitted model object (typically an `lmer` or `lm` object) created by `ANOVA_DCt()`, `ANOVA_DDCt()`.
`specs`	A character string or character vector specifying the predictors or combinations of predictors over which relative expression values are desired. This argument follows the specification syntax used by `emmeans::emmeans()` (e.g., `"Factor"`, `"Factor1 | Factor2"`).
`p.adj`	Character string specifying the method for adjusting p-values. See `p.adjust` for available options.

Details

The `Means_DDCt` function performs pairwise comparisons of relative expression values fo all combinations using estimated marginal means derived from a fitted model. For ANOVA models, relative expression values can be obtained for main effects, interactions, and sliced (simple) effects. For ANCOVA models returned by the **rtpcr** package, only simple effects are supported.

Internally, this function relies on the **emmeans** package to compute marginal means and contrasts, which are then back-transformed to fold change values using the ddCt framework.

Value

A data frame containing estimated relative expression values, confidence intervals, p-values, and significance levels derived from the fitted model.

Author(s)

Ghader Mirzaghaderi

Examples

```
data <- read.csv(system.file("extdata", "data_3factor.csv", package = "rtpcr"))

# Obtain a fitted model from ANOVA_DDCt
res <- ANOVA_DDCt(
  data,
  numOfFactors = 3,
  numberOfrefGenes = 1,
  specs = "Type",
  block = NULL)

# Relative expression values for Type main effect
lm <- res$perGene$PO$lm
Means_DDCt(lm, specs = "Type")

# Relative expression values for Concentration main effect
Means_DDCt(lm, specs = "Conc")

# Relative expression values for Concentration sliced by Type
Means_DDCt(lm, specs = "Conc | Type")

# Relative expression values for Concentration sliced by Type and SA
Means_DDCt(lm, specs = "Conc | Type * SA")
```

`meanTech` *Computing the mean of technical replicates*

Description

Computes the arithmetic mean of technical replicates for each sample or group. This is often performed before ANOVA or other statistical analyses to simplify comparisons between experimental groups.

Usage

```
meanTech(
  x,
  groups,
  numOfFactors,
  numberOfrefGenes,
  block,
  set_missing_target_Ct_to_40 = FALSE
)
```

Arguments

`x`	A raw data frame containing technical replicates.
`groups`	An integer vector or character vector specifying the column(s) to group by before calculating the mean of technical replicates.
`numOfFactors`	Integer. Number of experimental factor columns
`numberOfrefGenes`	Integer. Number of reference genes.
`block`	Character. Block column name or `NULL`.
`set_missing_target_Ct_to_40`	If `TRUE`, missing target gene Ct values become 40; if `FALSE` (default), they become NA.

Details

The `meanTech` function calculates the mean of technical replicates based on one or more grouping columns. This reduces the dataset to a single representative value per group, facilitating downstream analysis such as fold change calculation or ANOVA.

Value

A data frame with the mean of technical replicates for each group.

Author(s)

Ghader Mirzaghaderi

Examples

```
# Example input data frame with technical replicates
data1 <- read.csv(system.file("extdata", "data_withTechRep.csv", package = "rtpcr"))
```

```
# Calculate mean of technical replicates using first four columns as groups
meanTech(data1,
         groups = 1:2,
         numOfFactors = 1,
         numberOfrefGenes = 1,
         block = NULL)

# Another example using different dataset and grouping columns
data2 <- read.csv(system.file("extdata", "data_Lee_etal2020qPCR.csv", package = "rtpcr"))
meanTech(data2, groups = 1:3,
         numOfFactors = 2,
         numberOfrefGenes = 1,
         block = NULL)
```

`multiplot` *Combine multiple ggplot objects into a single layout*

Description

The `multiplot` function arranges multiple `ggplot2` objects into a single plotting layout with a specified number of columns.

Usage

```
multiplot(..., cols = 1)
```

Arguments

`...`	One or more `ggplot2` objects.
`cols`	Integer specifying the number of columns in the layout.

Details

Multiple `ggplot2` objects can be provided either as separate arguments via `...`. The function uses the `grid` package to control the layout.

Value

A grid object displaying multiple plots arranged in the specified layout.

Author(s)

Pedro J. (adapted from https://gist.github.com/pedroj/ffe89c67282f82c1813d)

Examples

```
# Example using output from TTEST_DDCt
data1 <- read.csv(system.file("extdata", "data_ttest18genes.csv", package = "rtpcr"))
out <- TTEST_DDCt(
  data1,
  paired = FALSE,
  var.equal = TRUE,
  numberOfrefGenes = 1)

p1 <- plotFactor(out,
  x_col = "gene",
  y_col = "log2FC",
  Lower.se_col = "Lower.se.log2FC",
  Upper.se_col = "Upper.se.log2FC",
  letters_col = "sig")

p2 <- plotFactor(out,
  x_col = "gene",
  y_col = "RE",
  Lower.se_col = "Lower.se.RE",
  Upper.se_col = "Upper.se.RE",
  letters_col = "sig")



# Example using output from ANOVA_DCt
data2 <- read.csv(system.file("extdata", "data_1factor.csv", package = "rtpcr"))
out2 <- ANOVA_DCt(
  data2,
  numOfFactors = 1,
  numberOfrefGenes = 1,
  block = NULL)

df <- out2$relativeExpression

p3 <- plotFactor(
  df,
  x_col = "SA",
  y_col = "RE",
  Lower.se_col = "Lower.se.RE",
  Upper.se_col = "Upper.se.RE",
  letters_col = "sig",
  letters_d = 0.1,
  col_width = 0.7,
  err_width = 0.15,
  fill_colors = "skyblue",
  alpha = 1,
  base_size = 14)

# Combine plots into a single layout
multiplot(p1, p2, cols = 2)
```

```
multiplot(p1, p3, cols = 2)
```

---

plotFactor *Bar plot of gene(s) expression for 1-, 2-, or 3-factor experiments*

Description

Creates a bar plot of relative gene expression (fold change) values from 1-, 2-, or 3-factor experiments, including error bars and statistical significance annotations.

Usage

```
plotFactor(
  data,
  x_col,
  y_col,
  Lower.se_col,
  Upper.se_col,
  group_col = NULL,
  facet_col = NULL,
  letters_col = NULL,
  letters_d = 0.2,
  col_width = 0.8,
  err_width = 0.15,
  dodge_width = 0.8,
  fill_colors = NULL,
  color = "black",
  alpha = 1,
  base_size = 12,
  legend_position = "right",
  removeCalibratorCols = FALSE,
  removeCalibratorText = FALSE,
  ...
)
```

Arguments

data	Data frame containing expression results
x_col	Character. Column name for x-axis
y_col	Character. Column name for bar height
Lower.se_col	Character. Column name for lower SE
Upper.se_col	Character. Column name for upper SE
group_col	Character. Column name for grouping bars (optional)
facet_col	Character. Column name for faceting (optional)
letters_col	Character. Column name for significance letters (optional)

letters_d	Numeric. Vertical offset for letters (default 0.2)
col_width	Numeric. Width of bars (default 0.8)
err_width	Numeric. Width of error bars (default 0.15)
dodge_width	Numeric. Width of dodge for grouped bars (default 0.8)
fill_colors	Optional vector of fill colors to change the default colors
color	Optional color for the bar outline
alpha	Numeric. Transparency of bars (default 1)
base_size	Numeric. Base font size for theme (default 12)
legend_position	Character or numeric vector. Legend position (default right)
removeCalibratorCols	= NULL or remove Calibrator Cols
removeCalibratorText	= NULL or remove Calibrator text
...	Additional ggplot2 layer arguments

Value

ggplot2 plot object

Author(s)

Ghader Mirzaghaderi

Examples

```
data <- read.csv(system.file("extdata", "data_2factorBlock3ref.csv", package = "rtpcr"))

res <- ANOVA_DDCt(x = data,
  numOfFactors = 2,
  numberOfrefGenes = 3,
  block = "block",
  specs = "Concentration",
  p.adj = "none")

df <- res$relativeExpression

p1 <- plotFactor(
  data = df,
  x_col = "contrast",
  y_col = "RE",
  group_col = "gene",
  facet_col = "gene",
  Lower.se_col = "Lower.se.RE",
  Upper.se_col = "Upper.se.RE",
  letters_col = "sig",
  letters_d = 0.2,
  alpha = 1,
```

```
  col_width = 0.7,
  dodge_width = 0.7,
  base_size = 14,
  legend_position = "none")

p1


data2 <- read.csv(system.file("extdata", "data_3factor.csv", package = "rtpcr"))
#Perform analysis first
res <- ANOVA_DCt(
  data2,
  numOfFactors = 3,
  numberOfrefGenes = 1,
  block = NULL)

df <- res$relativeExpression
# Generate three-factor bar plot
 p <- plotFactor(
  df,
  x_col = "SA",
  y_col = "log2FC",
  group_col = "Type",
  facet_col = "Conc",
  Lower.se_col = "Lower.se.log2FC",
  Upper.se_col = "Upper.se.log2FC",
  letters_col = "sig",
  letters_d = 0.3,
  col_width = 0.7,
  dodge_width = 0.7,
  #fill_colors = c("blue", "brown"),
  color = "black",
  base_size = 14,
  alpha = 1,
  legend_position = c(0.1, 0.2))
p
```

TTEST_DDCt *Delta Delta Ct method t-test analysis*

Description

The TTEST_DDCt function performs fold change expression analysis based on the $\Delta\Delta C_T$ method using Student's t-test. It supports analysis of one or more target genes evaluated under two experimental conditions (e.g. control vs treatment).

Usage

```
TTEST_DDCt(
```

```
  x,
  numberOfrefGenes,
  Factor.level.order = NULL,
  paired = FALSE,
  var.equal = TRUE,
  p.adj = "none",
  set_missing_target_Ct_to_40 = FALSE
)
```

Arguments

`x` A data frame containing experimental conditions, biological replicates, and amplification efficiency and Ct values for target and reference genes. The number of biological replicates must be equal across genes. If this is not true, or there are `NA` values use `ANODA_DDCt` function for independent samples or `REPEATED_DDCt` for paired samples. See the package vignette for details on the required data structure.

`numberOfrefGenes` Integer specifying the number of reference genes used for normalization.

`Factor.level.order` Optional character vector specifying the order of factor levels. If `NULL`, the first level of the factor column is used as the calibrator.

`paired` Logical; if `TRUE`, a paired t-test is performed.

`var.equal` Logical; if `TRUE`, equal variances are assumed and a pooled variance estimate is used. Otherwise, Welch's t-test is applied.

`p.adj` Method for p-value adjustment. One of `"holm"`, `"hochberg"`, `"hommel"`, `"bonferroni"`, `"BH"`, `"BY"`, `"fdr"`, or `"none"`. See `p.adjust`.

`set_missing_target_Ct_to_40` If `TRUE`, missing target gene Ct values become 40; if `FALSE` (default), they become NA.

Details

Relative expression values are computed using one or more reference genes for normalization. Both paired and unpaired experimental designs are supported.

Paired samples in quantitative PCR refer to measurements collected from the same individuals under two different conditions (e.g. before vs after treatment), whereas unpaired samples originate from different individuals in each condition. Paired designs allow within-individual comparisons and typically reduce inter-individual variability.

The function returns numerical summaries as well as bar plots based on either relative expression (RE) or log2 fold change (log2FC).

All the functions for relative expression analysis (including 'TTEST_DDCt()', 'WILCOX_DDCt()', 'ANOVA_DDCt()', 'ANCOVA_DDCt()', 'REPEATED_DDCt()', and 'ANOVA_DCt()') return the relative expression table which include fold change and corresponding statistics. The output of 'ANOVA_DDCt()', 'ANCOVA_DDCt()', 'ANCOVA_DDCt()', 'REPEATED_DDCt()', and 'ANOVA_DCt()' also include lm models, residuals, raw data and ANOVA table for each gene.

The expression table returned by 'TTEST_DDCt()', 'WILCOX_DDCt()', 'ANOVA_DDCt()', 'ANCOVA_DDCt()', and 'REPEATED_DDCt()' functions include these columns: gene (name of target genes), contrast (calibrator level and contrasts for which the relative expression is computed), RE (relative expression or fold change), log2FC (log(2) of relative expression or fold change), pvalue, sig (per-gene significance), LCL (95% lower confidence level), UCL (95% upper confidence level), se (standard error of mean calculated from the weighted delta Ct values of each of the main factor levels), Lower.se.RE (The lower limit error bar for RE which is 2^(log2(RE) - se)), Upper.se.RE (The upper limit error bar for RE which is 2^(log2(RE) + se)), Lower.se.log2FC (The lower limit error bar for log2 RE), and Upper.se.log2FC (The upper limit error bar for log2 RE)

Value

A list with the following components:

Result Table containing RE values, log2FC, p-values, significance codes, confidence intervals, standard errors, and lower/upper SE limits.

Author(s)

Ghader Mirzaghaderi

References

LivakKJ, Schmittgen TD (2001). Analysis of Relative Gene Expression Data Using Real-Time Quantitative PCR and the Double Delta CT Method. *Methods*, 25(4), 402–408. doi:10.1006/meth.2001.1262

Ganger MT, Dietz GD, and Ewing SJ (2017). A common base method for analysis of qPCR data and the application of simple blocking in qPCR experiments. *BMC Bioinformatics*, 18, 1–11.

Taylor SC, Nadeau K, Abbasi M, Lachance C, Nguyen M, Fenrich, J. (2019). The ultimate qPCR experiment: producing publication quality, reproducible data the first time. *Trends in Biotechnology*, 37, 761-774.

Yuan JS, Reed A, Chen F, Stewart N (2006). Statistical Analysis of Real-Time PCR Data. *BMC Bioinformatics*, 7, 85.

Examples

```
# Example data structure
data1 <- read.csv(system.file("extdata", "data_ttest18genes.csv", package = "rtpcr"))

# Unpaired t-test
TTEST_DDCt(
  data1,
  paired = FALSE,
  var.equal = TRUE,
  numberOfrefGenes = 1)

# With amplification efficiencies
data2 <- read.csv(system.file("extdata", "data_1factor_one_ref_Eff.csv", package = "rtpcr"))

TTEST_DDCt(
  data2,
```

```
  numberOfrefGenes = 1)

# Two reference genes
data3 <- read.csv(system.file("extdata", "data_1factor_Two_ref.csv", package = "rtpcr"))
TTEST_DDCt(
  data3,
  numberOfrefGenes = 2)
```

WILCOX_DDCt *Delta Delta Ct method wilcox.test analysis*

Description

The WILCOX_DDCt function performs fold change expression analysis based on the $\Delta\Delta C_T$ method using wilcox.test. It supports analysis of one or more target genes evaluated under two experimental conditions (e.g. control vs treatment).

Usage

```
WILCOX_DDCt(
  x,
  numberOfrefGenes,
  Factor.level.order = NULL,
  paired = FALSE,
  p.adj = "none",
  set_missing_target_Ct_to_40 = FALSE
)
```

Arguments

x
: A data frame containing experimental conditions, biological replicates, and amplification efficiency and Ct values for target and reference genes. The number of biological replicates must be equal across genes. If this is not true, or there are NA values use ANODA_DDCt function for independent samples or REPEATED_DDCt for paired samples. See the package vignette for details on the required data structure.

numberOfrefGenes
: Integer specifying the number of reference genes used for normalization.

Factor.level.order
: Optional character vector specifying the order of factor levels. If NULL, the first level of the factor column is used as the calibrator.

paired
: Logical; if TRUE, a paired wilcox.test is performed.

p.adj
: Method for p-value adjustment. One of "holm", "hochberg", "hommel", "bonferroni", "BH", "BY", "fdr", or "none". See p.adjust.

set_missing_target_Ct_to_40
: If TRUE, missing target gene Ct values become 40; if FALSE (default), they become NA.

Details

Relative expression values are computed using reference gene(s) for normalization. Both paired and unpaired experimental designs are supported.

Paired samples in quantitative PCR refer to measurements collected from the same individuals under two different conditions (e.g. before vs after treatment), whereas unpaired samples originate from different individuals in each condition. Paired designs allow within-individual comparisons and typically reduce inter-individual variability.

The function returns expression table. The expression table returned by 'TTEST_DDCt()', 'WILCOX_DDCt()', 'ANOVA_DDCt()', 'ANCOVA_DDCt()', and 'REPEATED_DDCt()' functions include these columns: gene (name of target genes), contrast (calibrator level and contrasts for which the relative expression is computed), RE (relative expression or fold change), log2FC (log(2) of relative expression or fold change), pvalue, sig (per-gene significance), LCL (95% lower confidence level), UCL (95% upper confidence level), se (standard error of mean calculated from the weighted delta Ct values of each of the main factor levels), Lower.se.RE (The lower limit error bar for RE which is 2^(log2(RE) - se)), Upper.se.RE (The upper limit error bar for RE which is 2^(log2(RE) + se)), Lower.se.log2FC (The lower limit error bar for log2 RE), and Upper.se.log2FC (The upper limit error bar for log2 RE)

Value

A table containing RE values, log2FC, p-values, significance, confidence intervals, standard errors, and lower/upper SE limits.

Author(s)

Ghader Mirzaghaderi

References

Yuan, J. S., Reed, A., Chen, F., and Stewart, N. (2006). Statistical Analysis of Real-Time PCR Data. *BMC Bioinformatics*, 7, 85.

Examples

```
# Example data structure
data <- read.csv(system.file("extdata", "data_Yuan2006PMCBioinf.csv", package = "rtpcr"))

# Unpaired t-test
WILCOX_DDCt(
  data,
  paired = FALSE,
  numberOfrefGenes = 1)


# Two reference genes
data2 <- read.csv(system.file("extdata", "data_1factor_Two_ref.csv", package = "rtpcr"))
WILCOX_DDCt(
  data2,
  numberOfrefGenes = 2,
  p.adj = "none")
```

Index

∗ **external**
data_2factorBlock3ref, 9

ANOVA_DCt, 2
ANOVA_DDCt, 4

compute_wDCt, 8

data_2factorBlock3ref, 9

efficiency, 10

long_to_wide, 11

Means_DDCt, 12
meanTech, 13
multiplot, 15

p.adjust, *3*, *5*, *12*, *20*, *22*
plotFactor, 17

TTEST_DDCt, 19

WILCOX_DDCt, 22