Package 'ddCt'

July 9, 2025

Title The ddCt Algorithm for the Analysis of Quantitative Real-Time PCR (qRT-PCR)

Version 1.64.0

Date 2021-07-30

Author Jitao David Zhang, Rudolf Biczok, and Markus Ruschhaupt

Description The Delta-Delta-Ct (ddCt) Algorithm is an approximation method to determine relative gene expression with quantitative real-time PCR (qRT-PCR) experiments. Compared to other approaches, it requires no standard curve for each primer-target pair, therefore reducing the working load and yet returning accurate enough results as long as the assumptions of the amplification efficiency hold. The ddCt package implements a pipeline to collect, analyse and visualize qRT-PCR results, for example those from TaqMan SDM software, mainly using the ddCt method. The pipeline can be either invoked by a script in command-line or through the API consisting of S4-Classes, methods and functions.

Maintainer Jitao David Zhang <jitao_david.zhang@roche.com>

biocViews GeneExpression, DifferentialExpression, MicrotitrePlateAssay, qPCR

License LGPL-3

Depends R (>= 2.3.0), methods

Imports Biobase (>= 1.10.0), RColorBrewer (>= 0.1-3), xtable, lattice, BiocGenerics

Collate AllGlobals.R AllClasses.R AllGenerics.R AllFunctions.R AllMethods.R absoluteQ.R ddCtSplit.R output.R visualization.R

Suggests testthat (>= 3.0.0), RUnit

Config/testthat/edition 3

git_url https://git.bioconductor.org/packages/ddCt

git_branch RELEASE_3_21

git_last_commit ed9b832

git_last_commit_date 2025-04-15

Repository Bioconductor 3.21

Date/Publication 2025-07-09

Contents

`barploterrbar` *Barplot with error bars.*

Description

Barplot with error bars.

Usage

```
barploterrbar(y, yl, yh, barcol="orange", errcol="black", horiz=FALSE,
w=0.2,theCut=NULL,columnForDiffBars=TRUE,cex.axis =
par("cex.axis"),zeroForNA=TRUE,legend=FALSE,groups = NULL, order=FALSE, ...)
```

Arguments

`y`	Numeric vector.
`yl`	Numeric vector of same length as y.
`yh`	Numeric vector of same length as y.
`barcol`	Color of the bars.
`errcol`	Color of the error bars.
`horiz`	Logical. As in `barplot`.
`w`	Size of the error bar ticks.
`theCut`	The cut value

columnForDiffBars	Whether the matrix should be transposed (by default the rows are for diff bars)
zeroForNA	Draw 0 instead of NA
cex.axis	Axis font cex
legend	Sould a legend be plotted ?
groups	a factor - if specified the bars are collored according to the group they belong to
order	plot sample values in descending order
...	Further arguments that get passed on to barplot.

Details

The function calls barplot with y and decorates it with error bars according to yl and yh.

Value

The function is called for its side effect, producing a plot.

Author(s)

Markus Ruschhaupt, Florian Hahne

See Also

barplot

Examples

```
y <- matrix(runif(80), ncol=5)
ym <- apply(y, 2, mean)
dy <- apply(y, 2, sd)*2/sqrt(nrow(y))
barploterrbar(ym, ym-dy, ym+dy, barcol="#0000c0", errcol="orange")
```

---

ddCtAbsolute *absolute quantification for Taqman data*

Description

absolute quantification for Taqman data

Usage

```
ddCtAbsolute(raw.table, addData, type = "mean", ADD = -30.234, DIV = -1.6268, sampleInformation = NULL,
```

Arguments

raw.table	data frame. It must contain columns with the following names:'Ct','Sample','Detector','Platename'. The column 'Ct' must contain numeric values.
addData	add data
type	character of length 1. 'mean' or 'median'- which method should be used for the aggregation of the repicates
ADD	Add constant
DIV	Div constant
sampleInformation	if specified it must be an object of class phenoData with a column named 'Sample'.
toZero	boolean - if there is only one replication should the error be treated as zero ? (only if 'type' is mean)
filename	character of length 1. The name of the file the warnings should be stored in.

Value

A an object of class eSet. The assayData has the following components: exprs, error, Ct, Ct.error, Difference, number_NA, number, Plate.

Author(s)

Markus Ruschhaupt mailto:m.ruschhaupt@dkfz.de

References

~put references to the literature/web site here ~

---

ddCtExpression-class *ddCt Expression*

Description

This class is a subclass of ExpressionSet and represents objects which are produced by the ddCt algorithm in the ddCtExpression method

Extends

Class ExpressionSet, directly. Class eSet, by class "ExpressionSet", distance 2. Class VersionedBiobase, by class "ExpressionSet", distance 3. Class Versioned, by class "ExpressionSet", distance 4.

Methods

Ct `signature(object = "ddCtExpression")`: returns the Ct value of this ddCtExpressionobject

CtErr `signature(object = "ddCtExpression")`: returns the error number of the Ct value of this ddCtExpressionobject

dCt `signature(object = "ddCtExpression")`: returns the dCt value of this ddCtExpressionobject

dCtErr `signature(object = "ddCtExpression")`: returns the error number of the dCt value of this ddCtExpressionobject

ddCt `signature(object = "ddCtExpression")`:returns the ddCt value of this ddCtExpressionobject

ddCtErr `signature(object = "ddCtExpression")`: returns the error number of the ddCt value of this ddCtExpressionobject

level `signature(object = "ddCtExpression")`: returns the levels in this ddCtExpressionobject

levelErr `signature(object = "ddCtExpression")`: returns the error number of the levens in this ddCtExpressionobject

numberCt `signature(object = "ddCtExpression")`: returns the Ct number of this ddCtExpressionobject

numberNA `signature(object = "ddCtExpression")`: returns the NA number of this ddCtExpressionobject

elist `signature(object = "ddCtExpression")`: returns a data frame which represents this expression object

elistWrite `signature(object = "ddCtExpression", file = "character")`: writes ddCtExpression object into a file

Author(s)

Rudolf Biczok <mailto:r.biczok@dkfz.de>

See Also

`SDMFrame`: reader for SDM files `elist`, `elistWrite`: utility functions for ddCtExpression objects `ddCtExpression`: the method which invokes the ddCt algorithm

Examples

```
## read a SDM file
sampdat <- SDMFrame(system.file("extdata", "Experiment1.txt",
                                package="ddCt"))

## call ddCtExpression method to get a ddCt calculated expression
result <- ddCtExpression(sampdat,
                         calibrationSample="Sample1",
                         housekeepingGenes=c("Gene1","Gene2"))

## use getter methods
ddCt(result)
ddCtErr(result)
```

ddCtExpression-methods

Apply the ddCt algorithm for a given data set

Description

Apply the ddCt algorithm for a given data set

Arguments

object	SDMFrame Data object which holds a data set containing columns with the following names: 'Ct','Sample','Detector','Platename'. The column 'Ct' must contain numeric values.
algorithm	character. Name of the calibration samples.
warningStream	character of length 1. The name of the file the warnings should be stored in.
calibrationSample	character. Name of the calibration samples.
housekeepingGenes	character. Name of the housekeeping genes.
type	character of length 1. 'mean' or 'median'- which method should be used for the aggregation of the repicates
sampleInformation	if specified it must be an object of class phenoData with a column named 'Sample'.
toZero	boolean - if there is only one replication should the error be treated as zero ? (only if 'type' is mean)
efficiencies	n.V.
efficiencies.error	n.V.

Value

A an object of class ddCtExpression.

usage

ddCtExpression(object, warningStream = "warning.output.txt", algorithm="ddCt" calibrationSample, housekeepingGenes, type="mean", sampleInformation=NULL, toZero=TRUE, efficiencies = NULL, efficiencies.error = NULL)

Methods

object = "InputFrame" An object of InputFrame, constructed with the method InputFrame

Author(s)

Rudolf Biczok <mailto:r.biczok@dkfz.de>

References

Analysis of relative gene expression data using real-time quantitative PCR and the 2(-Delta -Delta C(T)) Method. KJ Livak and TD Schmittgen, Methods, Vol. 25, No. 4. (December 2001), pp. 402-408

See Also

InputFrame: reader for SDM files ddCtExpression: representation for ddCt calculated expressions

Examples

```
## read a SDM file
sampdat <- SDMFrame(system.file("extdata", "Experiment1.txt", package="ddCt"))

## call ddCtExpression method from class SDMFrame
## to get a ddCt calculated expression
result <- ddCtExpression(sampdat,
                         calibrationSample="Sample1",
                         housekeepingGenes=c("Gene1","Gene2"))
result
```

---

elistWrite-methods *Write ddCtExpression object into data frame or files*

Description

ddCtExpression object contains a list of matrices as the results of ddCt method. elist combines these lists into one data frame, and elistWrite writes the data frame into file.

summary is a wrapper for the elist method

Usage

```
elist(object,...)
summary(object,...)
elistWrite(object,file,...)
```

Arguments

object	an ExpressionSet object.
file	output file.
...	additional arguments passed to write.table.

Details

`elist` is a wrapper to `as(object, "data.frame")` function.

Value

A data frame or output file.

Author(s)

Jitao David Zhang <jitao_david.zhang@roche.com>

Examples

```
## read a SDM file
sampdat <- SDMFrame(system.file("extdata", "Experiment1.txt", package="ddCt"))

## call ddCtExpression method from class SDMFrame
## to get a ddCt calculated expression
result <- ddCtExpression(sampdat,
                         calibrationSample="Sample1",
                         housekeepingGenes=c("Gene1","Gene2"))

## call elist
elistResult <- elist(result)
elistResult
```

`errBarchart-methods` *Draw barchart of relative expression level with error-bars*

Description

Draw barchart (with error-bars) of relative expression level represented in `ddCtExpression` object. The barchart is implemented as grid plot by `lattice` package, where each panel represents one sample and the relative expression values of detectors (as well as their standard errors) are depicted as bars.

Detectors which are not determined are marked by grey ND.

Two types of figures are supported: either condition on samples (by="Sample") or on detectors (by="Detector").

Methods

object = "ddCtExpression", by="character" An object of `ddCtExpression`, constructed with the method `ddCtExpression`

errBarchartParameter-class
Class "errBarchartParameter"

Description

Parameter object for errBarchart

Objects from the Class

Objects can be created by calls of the form `new("errBarchartParameter", ...)`. So far the object is only internally used, but in the near future it will be exported.

Slots

`exprsUndeterminedLabel`: Object of class `"character"`, specifying the text label when the expression level is 'Undetermined'

Methods

exprsUndeterminedLabel `signature(object = "errBarchartParameter")`: getting the text label when the expression level is 'Undetermined'

show `signature(object = "errBarchartParameter")`: print method

Note

So far it is only internallly used

Author(s)

Jitao David Zhang <jitao_david.zhang@roche.com>

Examples

```
## Internally used
## param <- new("errBarchartParameter")
## exprsUndeterminedLabel(param)
```

getDir *Auxillary functions for the executive scripts*

Description

getDir creates a directory in case it does not exist and returns the directory name.

Usage

```
getDir(dir, ...)
```

Arguments

dir	Directory name
...	Other parameters passed to dir.create

Details

Auxillary functions

Value

getDir returns the directory name

Author(s)

Jitao David Zhang <jitao_david.zhang@roche.com>

Examples

```
getDir(tempdir())
```

InputFrame *Build an InputFrame from a ReaderClass or a data frame*

Description

Generally an InputFrame is built from a ReaderClass (e.g. InputReader), or a data.frame. See the example below for building an object from a valid data.frame.

Usage

```
InputFrame(object)
```

Arguments

object	A data.frame with three columns: Sample, Detector, and Ct

Value

A object of class InputFrame

Author(s)

Jitao David Zhang mailto:jitao_david.zhang@roche.com

Examples

```
testDf <- data.frame(Sample=rep(paste("Sample", 1:3), each=2),
Detector=rep(paste("Gene", 1:2), 3),
Ct=30+rnorm(6))
testInputFrame <- InputFrame(testDf)
```

InputFrame-class *Working with SDM files*

Description

The class InputFrame provides core functionalities to read gene and sample information from SDM files and calculate them with a ddCt algorithm.

The function InputFrame reads the data given in the colums 'Detector','Sample' and 'Ct' of the specified SDM output files and stores them as a data.frame. An additional column including the respective filename is added.

Slots

coreData: Object of class "data.frame": Holds all the required data extracted from the SDM file

files: Object of class "character" contains the source SDM files

Methods

[,[[,\$ signature(x = "InputFrame"): primitive accessors. Returns an object of InputFrame-class with the subset data.

names signature(x = "InputFrame"): returns the column names in this SDM object

ddCtExpression signature(object = "InputFrame"): runs a ddCt algorithm with this SDM object and returns a object of class ddCtExpression

fileNames signature(object="InputFrame"): returns the source SDM file names.

detectorNames signature(object = "InputFrame"): returns the detector names in this SDM object

detectorNames<- signature(object = "InputFrame", value = "character"): replaces the detector names in this SDM object

sampleNames signature(object = "InputFrame"): returns the sample names in this SDM object

sampleNames<- `signature(object = "InputFrame", value = "character")`: replaces the sample names in this SDM object

uniqueDetectorNames `signature(object = "InputFrame")`: returns a vector of unique detector names in this SDM object

uniqueDetectorNames<- `signature(object = "InputFrame", target = "missing", value = "character")`: replaces all detector names given by the 'names' attribute in 'value' with new detector names

uniqueDetectorNames<- `signature(object = "InputFrame",target = "character", value = "character")`: replaces all detector names given by 'target' with new detector names

uniqueSampleNames<- `signature(object = "InputFrame", target = "missing", value = "character")`: replaces all sample names given by the 'names' attribute in 'value' with new sample names

uniqueSampleNames<- `signature(object = "InputFrame",target = "character", value = "character")`: replaces all sample names given by 'target' with new sample names

uniqueSampleNames `signature(object = "InputFrame")`: returns a vector of unique sample names in this SDM object

removeSample `signature(object = "InputFrame", sample="character")`: removes the sample(s) specified from the InputFrame object

replaceDetector `signature(object = "InputFrame",target="character", value="character")`: replace the detectors equal to the target with the value. Both `target` and `value` can be vectors of the same length, then the replace takes place iteratively.

replaceSample `signature(object = "InputFrame",target="character", value="character")`: replace the samples equal to the target with the value. Both `target` and `value` can be vectors of the same length, then the replace takes place iteratively.

show `signature(object="InputFrame")`: pretty print of the InputFrame instance.

rightCensoring `signature(object="InputFrame",threshold="numeric")`: Right censoring the Ct value, which targets the data points above a certain value (`threshold`). High Ct values (higher than 40 or 45 by the rule of thumb) are often not accurate and may indicate too weak expression. The function performs the right censoring on the data and set the value above the threshold as `NA` (by default) or a given value. See the example.

coreData `signature(object="InputFrame")`: returns the data frame read from SDM file.

coreData<- `signature(object="InputFrame")`: replace the data frame read from SDM file.

Ct `signature(object="InputFrame")`: returns the Ct value of the SDM file.

Ct `signature(object="InputFrame", value="numeric")`: replace the Ct value in the object with the new values, and return the object.

Author(s)

Rudolf Biczok <mailto:r.biczok@dkfz.de>, Jitao David Zhang <mailto:jitao_david.zhang@roche.com>

See Also

`SDMFrame` function reads in data from SDM files. Data from SDM files is used to construct `ddCtExpression` objects to analyze differetial expression.

Examples

```
## read a SDM file
sampdat <- SDMFrame(system.file("extdata", "Experiment1.txt",
                                package="ddCt"))
## you can also write
## sampdat <- new("SDMFrame",system.file("extdata", "Experiment1.txt",
##                                       package="ddCt"))

## use the getter methods
sampleNames(sampdat)

## or the overloaded primitive accessors
sampdat[1:3,"Sample"]

## see all unique samples
uniqueSampleNames(sampdat)

## replace all sample names 'Sample1' and 'Sample2' in sampdat
## with 'NewSample1' and 'NewSample2'
uniqueSampleNames(sampdat,c("Sample1","Sample2")) <- c("NewSample1","NewSample2")
uniqueSampleNames(sampdat)

## or use this syntax to replace the gene names
uniqueDetectorNames(sampdat) <- c(Gene1="NewGene1", Gene2="NewGene2")
uniqueDetectorNames(sampdat)

## remove sample or detector
removeSample(sampdat, "Sample1")
removeDetector(sampdat, "Gene1")

## replace sample or detector
replaceSample(sampdat, "Sample1", "Sample0")
replaceDetector(sampdat, "Gene1", "PLCG1")

## right censoring the data
rightCensoring(sampdat, 35)
rightCensoring(sampdat, 35, 35)
```

InputReader-class *Class "InputReader"*

Description

Abstract factory for data input

Objects from the Class

A virtual Class: No objects may be created from it.

Slots

`files`: Input files

`colmap`: Column mapping

Author(s)

Rudolf Biczok and Jitao David Zhang

Examples

```
showClass("InputReader")
```

QuantStudioFrame *Read QuantStudio file(s)*

Description

Read QuantStudio file(s)

Usage

```
QuantStudioFrame(file)
readQuantStudio(file)
```

Arguments

`file`	Character vector of filenames

Details

This function reads the data given in the QuantStudio output files

Value

A object of class `SDMFrame`

Author(s)

Jitao David Zhang `mailto:jitao_david.zhang@roche.com`

Examples

```
sampdat <- QuantStudioFrame(system.file("extdata", c("QuantStudio_File1.txt", "QuantStudio_File2.txt"),package='

## use the getter methods
sampleNames(sampdat)

## or the overloaded primitive accessors
sampdat[1:3,"Sample"]

## see all unique samples
uniqueSampleNames(sampdat)

## replace all sample names 'Sample1' and 'Sample2' in sampdat
## with 'NewSample1' and 'NewSample2'
uniqueSampleNames(sampdat,c("Sample1","Sample2")) <- c("NewSample1","NewSample2")
uniqueSampleNames(sampdat)

## or use this syntax to replace the gene names
uniqueDetectorNames(sampdat) <- c(Hs00559368_m1="Gene1", Hs02576168_g1="Gene2")
uniqueDetectorNames(sampdat)
```

`removeNTC-methods` *Remove NTC samples*

Description

NTC stands for Non-template controls. This method remove the NTC samples from the input object.

Methods

`signature(object = "ddCtExpression")` An object hat has been analyzed with the `ddCt` method

`signature(object = "InputFrame")` An input object

`replaceVectorByEquality`
REPLACE ITEMS OF VECTOR BY EQUALITY

Description

The function replces (or updates) the items of a given vector by checking the equality with the `target` parameter. If found, the item will be replaced by the `value` parameter. The length of both `target` and `value` must be the same and could be longer than 1, in which case the replace will be iterated.

Usage

```
replaceVectorByEquality(vector, target, value)
```

Arguments

vector	A vector to be replaced. The items of the vector must be atom types, since the equality is checked by '=='.
target	targets to be replaced, could be either single or a vector
value	values to be replaced at the positions of targets, must be of the same length of target

Details

A warning will be prompted if any item in the target cannot be found

Value

A vector of the same length as the parameter vector

Author(s)

Jitao David Zhang

See Also

== for checking equality.

Examples

```
vector <- c("java", "perl", "python", "c#")
replaceVectorByEquality(vector, target="c#", value="c/c++")
replaceVectorByEquality(vector, target=c("c#","perl"), value=c("c/c++","R"))
```

SDMFrame *Read an SDM file*

Description

Read an SDM file: Data Output File for SDS, Version 2.1

Usage

```
SDMFrame(file)
readSDM(file)
```

Arguments

file	Character vector of filenames

Details

This function reads the data given in the colums 'Detector','Sample' and 'Ct' of the specified SDM output file(s) and stores them as a data.frame. An additional column including the respective filename is added.

This function is a wrapper for the SDMFrame constructor

Value

A object of class SDMFrame

Author(s)

Rudolf Biczok mailto:r.biczok@dkfz.de

Examples

```
## read a SDM file
sampdat <- SDMFrame(system.file("extdata", "Experiment1.txt",
                                package="ddCt"))
## you can also write
## sampdat <- new("SDMFrame",system.file("extdata", "Experiment1.txt",
##                                       package="ddCt"))

## or with
## sampdat <- readSDM(system.file("extdata", "Experiment1.txt",
##                                  package="ddCt"))

## use the getter methods
sampleNames(sampdat)

## or the overloaded primitive accessors
sampdat[1:3,"Sample"]

## see all unique samples
uniqueSampleNames(sampdat)

## replace all sample names 'Sample1' and 'Sample2' in sampdat
## with 'NewSample1' and 'NewSample2'
uniqueSampleNames(sampdat,c("Sample1","Sample2")) <- c("NewSample1","NewSample2")
uniqueSampleNames(sampdat)

## or use this syntax to replace the gene names
uniqueDetectorNames(sampdat) <- c(Gene1="NewGene1", Gene2="NewGene2")
uniqueDetectorNames(sampdat)
```

write.htmltable *Write a data frame into an html table within a html page*

Description

Write a 'data.frame' into an html table within a html page

Usage

```
write.htmltable(x, file, title = "", sortby = NULL, decreasing = TRUE, open = "wt")
```

Arguments

x	'data.frame'
file	character. File name.
title	character. Title of html page
sortby	character. Name of column by which to sort the table rows
decreasing	logical. Should the sort order be increasing or decreasing?
open	character. This argument is passed onto 'file'

Value

The funciton is called for its side effect: writing a file

Author(s)

Wolfgang Huber

writeSimpleTabCsv *Write a data frame into a tab delimited file*

Description

Write a 'data.frame' into a tab delimited file (not quoted and no-row-name TSV file)

Usage

```
writeSimpleTabCsv(x, file, ...)
```

Arguments

x	'data.frame'
file	character. File name.
...	Additional arguments passed onto the function

Value

The function is called for its side effect: writing a file

Author(s)

Wolfgang Huber

Index