Package 'EasyqpcR'

October 9, 2015

Type Package

Title EasyqpcR for low-throughput real-time quantitative PCR data analysis

Version 1.10.2

Date 2015-08-25

Author Le Pape Sylvain

Maintainer Le Pape Sylvain <sylvain.le.pape@univ-poitiers.fr>

Description This package is based on the qBase algorithms published by Hellemans et al. in 2007. The EasyqpcR package allows you to import easily qPCR data files as described in the vignette. Thereafter, you can calculate amplification efficiencies, relative quantities and their standard errors, normalization factors based on the best reference genes choosen (using the SLqPCR package), and then the normalized relative quantities, the NRQs scaled to your control and their standard errors. It proposes also a function to aggregate the qPCR biological replicates followed by the application of the algorithm published by Willems et al. (2008) which is a standardization procedure that can be applied to data sets that display high variations between biological replicates. This enables proper statistical analysis to draw relevant conclusions. This package has been created for low-throughput qPCR data analysis.

Imports plyr, matrixStats, plotrix, gWidgetsRGtk2

Suggests SLqPCR, qpcrNorm, qpcR, knitr

biocViews qPCR, GeneExpression

License GPL (>=2)

NeedsCompilation no

R topics documented:

`EasyqpcR-package` *Functions to analyse real-time quantitative PCR data at IRTOMIT-INSERM U1082*

Description

This package contains functions to analyse real-time quantitative PCR data at IRTOMIT-INSERM U1082. The algorithm used is the one published by Hellemans et al. (2007). It permits the calculation of the normalization and calibration factors, the relative quantities, the normalized relative quantities which can then be scaled to your control group.

Details

Package:	EasyqpcR
Type:	Package
Version:	1.1.3
Date:	2012-11-26
License:	GPL (>=2)

Author(s)

Sylvain Le Pape

Maintainer: Sylvain Le Pape <sylvain.le.pape@univ-poitiers.fr>

References

Jan Hellemans, Geert Mortier, Anne De Paepe, Frank Speleman and Jo Vandesompele. qBase relative quantification framework and software for management and automated analysis of real-time quantitative PCR data. Genome Biology 2007, 8:R19 (doi:10.1186/gb-2007-8-2-r19). <url:http://genomebiology.com/2007/8

badCt *Evaluation of the qPCR technical replicates*

Description

This function allows you to evaluate your qPCR technical replicates, you only need to define the threshold (according to Hellemans et al. (2007), 0.5 is a good threshold value), the dataset and the number of technical replicates you have done. I recommand you to use the gWidgets package to easily exclude the failed replicates.

Usage

```
badCt(data, r, threshold, na.rm = FALSE)
```

Arguments

`data` data.frame containing row datas (genes in columns, samples in rows, Cq values).

`r` numeric, number of qPCR replicates.

`threshold` numeric, the maximal variation between your qPCR replicates.

`na.rm` a logical value indicating whether NA values should be stripped before the computation proceeds.

Details

To facilitate the use of the function, I suggest you to use the gWidgets package as described in the vignette.

Value

This function returns the position (sample position and column position) where the variation between qPCR replicates is superior to the threshold value.

Author(s)

Sylvain Le pape <sylvain.le.pape@univ-poitiers.fr>

References

Jan Hellemans, Geert Mortier, Anne De Paepe, Frank Speleman and Jo Vandesompele. qBase relative quantification framework and software for management and automated analysis of real-time quantitative PCR data. Genome Biology 2007, 8:R19 (doi:10.1186/gb-2007-8-2-r19). <url:http://genomebiology.com/2007/8

Examples

```
data(qPCR_run1)

badCt(data=qPCR_run1, r=3, threshold=0.3, na.rm=TRUE)
```

calData *Calculation of calibration factors*

Description

This function determines the calibration factors (CF) using the method described in Hellemans et al. (2007).

Usage

```
calData(data)
```

Arguments

data	data.frame containing the NRQs of your calibrator(s) for each gene obtained by the nrmData function of this package.

Details

This function is necessary for comparing different quantitative real-time PCR runs to reduce the inter-run variability (Hellemans et al. (2007)). Then, the results obtained have to be included in an R object and then be inputed in the nrmData function (see the vignette for more informations).

Value

This function returns the calibration factor associated to each gene for the whole runs.

Author(s)

Sylvain Le Pape (IRTOMIT-INSERM U1082) <sylvain.le.pape@univ-poitiers.fr>

References

Jan Hellemans, Geert Mortier, Anne De Paepe, Frank Speleman and Jo Vandesompele. qBase relative quantification framework and software for management and automated analysis of real-time quantitative PCR data. Genome Biology 2007, 8:R19 (doi:10.1186/gb-2007-8-2-r19). <url:http://genomebiology.com/2007/8

Examples

```
data(qPCR_run1,qPCR_run2,qPCR_run3)

nrmData(data = qPCR_run1 , r=3, E=c(2, 2, 2, 2),
      Eerror=c(0.02, 0.02, 0.02, 0.02), nSpl=5,
      nbRef=2, Refposcol=1:2, nCTL=2,
      CF=c(1, 1, 1, 1), CalPos=5, trace=TRUE, geo=TRUE, na.rm=TRUE)

nrmData(data = qPCR_run2 , r=3, E=c(2, 2, 2, 2),
      Eerror=c(0.02, 0.02, 0.02, 0.02), nSpl=5,
      nbRef=2, Refposcol=1:2, nCTL=2,
      CF=c(1, 1, 1, 1), CalPos=5, trace=TRUE, geo=TRUE, na.rm=TRUE)

nrmData(data = qPCR_run3 , r=3, E=c(2, 2, 2, 2),
      Eerror=c(0.02, 0.02, 0.02, 0.02), nSpl=5,
      nbRef=2, Refposcol=1:2, nCTL=2,
      CF=c(1, 1, 1, 1), CalPos=5, trace=TRUE, geo=TRUE, na.rm=TRUE)


## Isolate the calibrator NRQ values of the first biological replicate

a <- nrmData(data = qPCR_run1 , r=3, E=c(2, 2, 2, 2),
      Eerror=c(0.02, 0.02, 0.02, 0.02), nSpl=5,
      nbRef=2, Refposcol=1:2, nCTL=2,
      CF=c(1, 1, 1, 1), CalPos=5, trace=TRUE, geo=TRUE, na.rm=TRUE)[[3]]

## Isolate the calibrator NRQ values of the first biological replicate

b <- nrmData(data = qPCR_run2 , r=3, E=c(2, 2, 2, 2),
      Eerror=c(0.02, 0.02, 0.02, 0.02), nSpl=5,
      nbRef=2, Refposcol=1:2, nCTL=2,
      CF=c(1, 1, 1, 1), CalPos=5, trace=TRUE, geo=TRUE, na.rm=TRUE)[[3]]

## Isolate the calibrator NRQ values of the first biological replicate

c <- nrmData(data = qPCR_run3 , r=3, E=c(2, 2, 2, 2),
      Eerror=c(0.02, 0.02, 0.02, 0.02), nSpl=5,
      nbRef=2, Refposcol=1:2, nCTL=2,
      CF=c(1, 1, 1, 1), CalPos=5, trace=TRUE, geo=TRUE, na.rm=TRUE)[[3]]

## Regrouping the calibrator NRQ values of all the biological replicates

d <- rbind(a, b, c)

## Calibration factor calculation

e <- calData(d)
```

`Efficiency_calculation` *Raw data for primer amplification efficiency calculation.*

Description

This is a dataset containg the raw data (Cq values) of a qPCR run for the primer amplification efficiency calculation.

Usage

```
data(Efficiency_calculation)
```

Format

A data frame with 15 observations on the following 3 variables.

`Samples` 5 cDNA dilutions (1/1, 1/10, 1/100, 1/1000, 1/10000)

`Gene.1` The first gene

`Gene.2` The second gene

Details

This data.frame is composed by 2 genes (Gene.1 and Gene.2). There are 5 cDNA dilutions (1/1, 1/10, 1/100, 1/1000, 1/10000).

Source

S. Le Pape, IRTOMIT-INSERM U1082.

References

S. Le Pape, IRTOMIT-INSERM U1082.

Examples

```
data(Efficiency_calculation)
```

`Gene_maximisation` *A data frame containing all the data from four different qPCR runs.*

Description

A data frame containing all the data from four different qPCR runs where samples and genes are spread accross runs.

Usage

```
data(Gene_maximisation)
```

Format

A data frame with 192 observations on the following 4 variables.

`Samples` a factor with levels `IRC 1.1` `IRC 1.2` `IRC 1.3` `IRC 1.4` `IRC 1.5` `IRC 1.6` `IRC 1.7` `IRC 1.8` `IRC 2.1` `IRC 2.2` `IRC 2.3` `IRC 2.4` `IRC 2.5` `IRC 2.6` `IRC 2.7` `IRC 2.8` `IRC 3.1` `IRC 3.2` `IRC 3.3` `IRC 3.4` `IRC 3.5` `IRC 3.6` `IRC 3.7` `IRC 3.8` `NTC 1` `NTC 2` `NTC 3` `NTC 4` `NTC 5` `NTC 6` `NTC 7` `NTC 8` `Sample 1` `Sample 10` `Sample 11` `Sample 12` `Sample 13` `Sample 14` `Sample 15` `Sample 16` `Sample 17` `Sample 18` `Sample 19` `Sample 2` `Sample 20` `Sample 21` `Sample 22` `Sample 23` `Sample 24` `Sample 25` `Sample 26` `Sample 27` `Sample 28` `Sample 29` `Sample 3` `Sample 30` `Sample 31` `Sample 32` `Sample 4` `Sample 5` `Sample 6` `Sample 7` `Sample 8` `Sample 9`

- `RG1` a numeric vector
- `RG2` a numeric vector
- `TG` a numeric vector

Details

This data frame is composed by 64 samples including IRCs (3 per gene), NTC (1 per gene). The control samples are Samples 1 to 16, and the test samples are Samples 17 to 32. There are 2 reference genes (RG1, RG2) and one target gene (TG).

Source

S. Le Pape, IRTOMIT-INSERM U1082.

References

S. Le Pape, IRTOMIT-INSERM U1082.

Examples

```
data(Gene_maximisation)
```

`Gene_maximisation_cor` *A data frame containing all the data from four different qPCR runs.*

Description

The Gene_maximisation data frame with 2 removed values (RG2, Sample 15, Ct = 22.06916 ; and RG3, Sample 18, Ct = 19.02328).

Usage

```
data(Gene_maximisation_cor)
```

Format

A data frame with 192 observations on the following 4 variables.

- `Samples` a factor with levels `IRC 1.1` `IRC 1.2` `IRC 1.3` `IRC 1.4` `IRC 1.5` `IRC 1.6` `IRC 1.7` `IRC 1.8` `IRC 2.1` `IRC 2.2` `IRC 2.3` `IRC 2.4` `IRC 2.5` `IRC 2.6` `IRC 2.7` `IRC 2.8` `IRC 3.1` `IRC 3.2` `IRC 3.3` `IRC 3.4` `IRC 3.5` `IRC 3.6` `IRC 3.7` `IRC 3.8` `NTC 1` `NTC 2` `NTC 3` `NTC 4` `NTC 5` `NTC 6` `NTC 7` `NTC 8` `Sample 1` `Sample 10` `Sample 11` `Sample 12` `Sample 13` `Sample 14` `Sample 15` `Sample 16` `Sample 17` `Sample 18` `Sample 19` `Sample 2` `Sample 20` `Sample 21` `Sample 22` `Sample 23` `Sample 24` `Sample 25` `Sample 26` `Sample 27` `Sample 28` `Sample 29` `Sample 3` `Sample 30` `Sample 31` `Sample 32` `Sample 4` `Sample 5` `Sample 6` `Sample 7` `Sample 8` `Sample 9`
- `RG1` a numeric vector
- `RG2` a numeric vector
- `TG` a numeric vector

Details

This data frame is composed by 64 samples including IRCs (3 per gene), NTC (1 per gene). The control samples are Samples 1 to 16, and the test samples are Samples 17 to 32. There are 2 reference genes (RG1, RG2) and one target gene (TG). This data frame is the Gene_maximisation data frame where 2 values have been removed (RG2, Sample 15, Ct = 22.06916 ; and RG3, Sample 18, Ct = 19.02328).

Source

S. Le Pape, IRTOMIT-INSERM U1082.

References

S. Le Pape, IRTOMIT-INSERM U1082.

Examples

```
data(Gene_maximisation_cor)
```

---

nrmData *Determination of the NF, RQ, NRQ, NRQ scaled to control and their SE and SD.*

Description

This function determines the values of the normalization factors, the relative quantitues, the normalized relative quantities, the normalized relative quantities scaled to control and their respectives standard errors and standard deviations by the method described by Hellemans et al. (2007).

Usage

```
nrmData(data, r, E, Eerror, nSpl, nbRef, Refposcol, nCTL, CF, CalPos,
trace = FALSE, geo = FALSE, na.rm = na.rm)
```

Arguments

`data`	data.frame containing row datas (genes in columns, samples in rows, Cq values).
`r`	numeric, number of qPCR replicates.
`E`	numeric, amplification efficiency values for each gene (follow the same order of the genes).
`Eerror`	numeric, standard errors of amplification efficiencies for each gene (follow the same order of the genes).
`nSpl`	numeric, number of samples to analyzed.
`nbRef`	numeric, number of reference genes used.
`Refposcol`	column position of your reference gene(s).

`nCTL` numeric, number of samples forming your control group.

`CF` numeric (or object if you have used the calData function from this package), values of the calibration factors for each gene (follow the same order of the genes).

`CalPos` numeric, sample number of your calibrator(s).

`trace` logical, print additional information.

`geo` logical, to scale to your control group, the function will use the geometrical mean if TRUE or the arithmetic mean if FALSE.

`na.rm` a logical value indicating whether NA values should be stripped before the computation proceeds.

Details

The algorithm used in this function is based on the article of Hellemans et al. (2007). This function calculates the expression value scaled to your control group and normalized to the calibration factor and the normalization factor. The limiting step is that you need to put the control samples on the top of the data frame otherwise, the algorithm will not work correctly. For more information for the way to use this function, please see the vignette.

Value

`NRQs normalized to control`
Gives the normlized relative quantities scaled to your control group.

`NRQs` Gives the normlized relative quantities.

`NRQs of your calibrator for this run`
Gives the normlized relative quantities of your calibrator(s).

Author(s)

Sylvain Le pape <sylvain.le.pape@univ-poitiers.fr>

References

Jan Hellemans, Geert Mortier, Anne De Paepe, Frank Speleman and Jo Vandesompele. qBase relative quantification framework and software for management and automated analysis of real-time quantitative PCR data. Genome Biology 2007, 8:R19 (doi:10.1186/gb-2007-8-2-r19). <url:http://genomebiology.com/2007/8

Examples

```
data(qPCR_run1,qPCR_run2,qPCR_run3)

nrmData(data = qPCR_run1 , r=3, E=c(2, 2, 2, 2),
      Eerror=c(0.02, 0.02, 0.02, 0.02), nSpl=5,
      nbRef=2, Refposcol=1:2, nCTL=2,
      CF=c(1, 1, 1, 1), CalPos=5, trace=TRUE, geo=TRUE, na.rm=TRUE)

nrmData(data = qPCR_run2 , r=3, E=c(2, 2, 2, 2),
      Eerror=c(0.02, 0.02, 0.02, 0.02), nSpl=5,
      nbRef=2, Refposcol=1:2, nCTL=2,
      CF=c(1, 1, 1, 1), CalPos=5, trace=TRUE, geo=TRUE, na.rm=TRUE)

nrmData(data = qPCR_run3 , r=3, E=c(2, 2, 2, 2),
      Eerror=c(0.02, 0.02, 0.02, 0.02), nSpl=5,
      nbRef=2, Refposcol=1:2, nCTL=2,
      CF=c(1, 1, 1, 1), CalPos=5, trace=TRUE, geo=TRUE, na.rm=TRUE)
```

qPCR_run1 *Raw data from the first qPCR run.*

Description

This is a dataset containg the raw data (Cq values) of the first qPCR run (the first biological replicate).

Usage

```
data(qPCR_run1)
```

Format

A data frame with 15 observations on the following 5 variables.

`Samples` a factor with levels `Calibrator` `Control 1` `Control 2` `Treatment 1` `Treatment 2`

`RG1` a numeric vector

`RG2` a numeric vector

`TG` a numeric vector

`TGb` a numeric vector

Details

This data.frame is composed by 2 reference genes (RG1 and RG2), and two target genes (TG and TGb). There are 5 samples, starting by the controls (1 and 2), the treatment groups (1 and 2), and a calibrator (calibrator). It is the first biological replicate and there is a qPCR triplicate by sample.

Source

S. Le Pape, IRTOMIT-INSERM U1082.

References

S. Le Pape, IRTOMIT-INSERM U1082.

Examples

```
data(qPCR_run1)
```

`qPCR_run2` *Raw data from the second qPCR run.*

Description

This is a dataset containg the raw data (Cq values) of the second qPCR run (the second biological replicate).

Usage

```
data(qPCR_run2)
```

Format

A data frame with 15 observations on the following 5 variables.

`Samples` a factor with levels `Calibrator` `Control 1` `Control 2` `Treatment 1` `Treatment 2`

`RG1` a numeric vector

`RG2` a numeric vector

`TG` a numeric vector

`TGb` a numeric vector

Details

This data.frame is composed by 2 reference genes (RG1 and RG2), and two target genes (TG and TGb). There are 5 samples, starting by the controls (1 and 2), the treatment groups (1 and 2), and a calibrator (calibrator). It is the second biological replicate and there is a qPCR triplicate by sample.

Source

S. Le Pape, IRTOMIT-INSERM U1082.

References

S. Le Pape, IRTOMIT-INSERM U1082.

Examples

```
data(qPCR_run2)
```

qPCR_run3 *Raw data from the third qPCR run.*

Description

This is a dataset containg the raw data (Cq values) of the third qPCR run (the third biological replicate).

Usage

```
data(qPCR_run3)
```

Format

A data frame with 15 observations on the following 5 variables.

`Samples` a factor with levels `Calibrator` `Control 1` `Control 2` `Treatment 1` `Treatment 2`

`RG1` a numeric vector

`RG2` a numeric vector

`TG` a numeric vector

`TGb` a numeric vector

Details

This data.frame is composed by 2 reference genes (RG1 and RG2), and two target genes (TG and TGb). There are 5 samples, starting by the controls (1 and 2), the treatment groups (1 and 2), and a calibrator (calibrator). It is the third biological replicate and there is a qPCR triplicate by sample.

Source

S. Le Pape, IRTOMIT-INSERM U1082.

References

S. Le Pape, IRTOMIT-INSERM U1082.

Examples

```
data(qPCR_run3)
```

slope *Function to calculate the amplification efficiency*

Description

This function calculates the amplification efficiency from classical qPCR dilution experiment using the Cq values. The Cq values are plotted against the logarithmized concentration (or dilution) values, a linear regression line is fit and the efficiency calculated by E = 10^-1/slope.

Usage

```
slope(data, q, r, na.rm = FALSE)
```

Arguments

`data`	data.frame containing row datas (genes in columns, samples in rows, Cq values).
`q`	numeric, cDNA dilution values.
`r`	numeric, number of qPCR replicates.
`na.rm`	logical, indicating whether NA values should be stripped before the computation proceeds.

Value

`Efficiency`	Primer amplification efficiency.
`Slope`	Slope of the dilution curve.
`Intercept`	Intercept of the dilution curve

Author(s)

Sylvain Le pape <sylvain.le.pape@univ-poitiers.fr>

See Also

You can also see the `qpcR` package with the `calib` and `calib2` functions.

Examples

```
data(Efficiency_calculation)

slope(data = Efficiency_calculation, q=c(1000, 100 ,10, 1, 0.1), r=3, na.rm=TRUE)
```

totData *Aggregation of qPCR biological replicates and data transformation*

Description

This function aggregates qPCR biological replicates and performs the algorithm published by Willems et al. (2008) which is a standardization procedure that can be applied to data sets that display high variation between biological replicates. This enables proper statistical analysis and drawing relevant conclusions. The procedure is not new, it has been used in microarray data analysis and is based on a series of sequential corrections, including log transformation, mean centering, and autoscaling.

Usage

```
totData(data, r, base, nSpl, nCTL, na.rm = na.rm)
```

Arguments

`data`	data.frame containing row datas (genes in columns, samples in rows, Cq values).
`r`	numeric, number of qPCR replicates.
`base`	numeric, the logarithmic base (2 or 10).
`nSpl`	numeric, the total number of samples (from all the biological replicates).
`nCTL`	numeric, number of samples forming your control group.
`na.rm`	logical, indicating whether NA values should be stripped before the computation proceeds.

Details

The standardization procedure used in this function (if TRUE for the transformation argument) is based on the article of Willems et al. (2008). This function performs successively these operations : log-transformation of your raw data, mean centering of the log-transformed data, autoscaling of your mean-centered and log-transformed data, i.e., your mean-centered data for each biological replicate will be divided by the standard deviation of the mean-centered biological replicate and then multiplicated by the mean of the standard deviation of all the biological replicates. The lower and upper confidence intervals (at 95%) are also computed.

For more information for the way to use this function, please see the vignette.

Value

`Standardized data according to Willems procedure (2008)`
The standardized data according to the Willems procedure (2008). These data must be used for statistical analysis.

`Mean` The mean of the biological replicates for each sample.

`Lower Confidence Interval 95%`
The Lower Confidence Interval 95% of the biological replicates for each sample.

`Upper Confidence Interval 95%`
The Upper Confidence Interval 95% of the biological replicates for each sample.

Author(s)

Sylvain Le pape <sylvain.le.pape@univ-poitiers.fr>

References

Erik Willems Luc Leyns, Jo Vandesompele. Standardization of real-time PCR gene expression data from independent biological replicates. Analytical Biochemistry 379 (2008) 127-129 (doi:10.1016/j.ab.2008.04.036). <url:http://www.sciencedirect.com/science/article/pii/S0003269708002649>

Examples

```
data(qPCR_run1,qPCR_run2,qPCR_run3)

nrmData(data = qPCR_run1 , r=3, E=c(2, 2, 2, 2),
        Eerror=c(0.02, 0.02, 0.02, 0.02), nSpl=5,
      nbRef=2, Refposcol=1:2, nCTL=2,
      CF=c(1, 1, 1, 1), CalPos=5, trace=TRUE, geo=TRUE, na.rm=TRUE)

nrmData(data = qPCR_run2 , r=3, E=c(2, 2, 2, 2),
      Eerror=c(0.02, 0.02, 0.02, 0.02), nSpl=5,
      nbRef=2, Refposcol=1:2, nCTL=2,
      CF=c(1, 1, 1, 1), CalPos=5, trace=TRUE, geo=TRUE, na.rm=TRUE)

nrmData(data = qPCR_run3 , r=3, E=c(2, 2, 2, 2),
      Eerror=c(0.02, 0.02, 0.02, 0.02), nSpl=5,
      nbRef=2, Refposcol=1:2, nCTL=2,
      CF=c(1, 1, 1, 1), CalPos=5, trace=TRUE, geo=TRUE, na.rm=TRUE)


## Isolate the calibrator NRQ values of the first biological replicate

a <- nrmData(data = qPCR_run1 , r=3, E=c(2, 2, 2, 2),
      Eerror=c(0.02, 0.02, 0.02, 0.02), nSpl=5,
      nbRef=2, Refposcol=1:2, nCTL=2,
      CF=c(1, 1, 1, 1), CalPos=5, trace=TRUE, geo=TRUE, na.rm=TRUE)[[3]]

## Isolate the calibrator NRQ values of the first biological replicate

b <- nrmData(data = qPCR_run2 , r=3, E=c(2, 2, 2, 2),
      Eerror=c(0.02, 0.02, 0.02, 0.02), nSpl=5,
      nbRef=2, Refposcol=1:2, nCTL=2,
      CF=c(1, 1, 1, 1), CalPos=5, trace=TRUE, geo=TRUE, na.rm=TRUE)[[3]]

## Isolate the calibrator NRQ values of the first biological replicate

c <- nrmData(data = qPCR_run3 , r=3, E=c(2, 2, 2, 2),
      Eerror=c(0.02, 0.02, 0.02, 0.02), nSpl=5,
      nbRef=2, Refposcol=1:2, nCTL=2,
      CF=c(1, 1, 1, 1), CalPos=5, trace=TRUE, geo=TRUE, na.rm=TRUE)[[3]]

## Regrouping the calibrator NRQ values of all the biological replicates
```

```
d <- rbind(a, b, c)

## Calibration factor calculation

e <- calData(d)


## Attenuation of inter-run variation thanks to the calibration factor for the
## first biological replicate

nrmData(data = qPCR_run1 , r=3, E=c(2, 2, 2, 2),
      Eerror=c(0.02, 0.02, 0.02, 0.02), nSpl=5,
      nbRef=2, Refposcol=1:2, nCTL=2,
      CF=e, CalPos=5, trace=TRUE, geo=TRUE, na.rm=TRUE)

## Attenuation of inter-run variation thanks to the calibration factor for the
## second biological replicate

nrmData(data = qPCR_run2 , r=3, E=c(2, 2, 2, 2),
      Eerror=c(0.02, 0.02, 0.02, 0.02), nSpl=5,
      nbRef=2, Refposcol=1:2, nCTL=2,
      CF=e, CalPos=5, trace=TRUE, geo=TRUE, na.rm=TRUE)

## Attenuation of inter-run variation thanks to the calibration factor for the
## third biological replicate

nrmData(data = qPCR_run3 , r=3, E=c(2, 2, 2, 2),
      Eerror=c(0.02, 0.02, 0.02, 0.02), nSpl=5,
      nbRef=2, Refposcol=1:2, nCTL=2,
      CF=e, CalPos=5, trace=TRUE, geo=TRUE, na.rm=TRUE)

## Isolate the NRQs scaled to control of the first biological replicate

a1 <- nrmData(data = qPCR_run1 , r=3, E=c(2, 2, 2, 2),
      Eerror=c(0.02, 0.02, 0.02, 0.02), nSpl=5,
      nbRef=2, Refposcol=1:2, nCTL=2,
      CF=e, CalPos=5, trace=TRUE, geo=TRUE, na.rm=TRUE)[1]

## Isolate the NRQs scaled to control of the second biological replicate

b1 <- nrmData(data = qPCR_run2 , r=3, E=c(2, 2, 2, 2),
      Eerror=c(0.02, 0.02, 0.02, 0.02), nSpl=5,
      nbRef=2, Refposcol=1:2, nCTL=2,
      CF=e, CalPos=5, trace=TRUE, geo=TRUE, na.rm=TRUE)[1]

## Isolate the NRQs scaled to control of the third biological replicate

c1 <- nrmData(data = qPCR_run3 , r=3, E=c(2, 2, 2, 2),
      Eerror=c(0.02, 0.02, 0.02, 0.02), nSpl=5,
      nbRef=2, Refposcol=1:2, nCTL=2,
      CF=e, CalPos=5, trace=TRUE, geo=TRUE, na.rm=TRUE)[1]

## Data frame transformation
```

```
a2 <- as.data.frame(a1)
b2 <- as.data.frame(b1)
c2 <- as.data.frame(c1)

## Aggregation of the three biological replicates

d2 <- rbind(a2, b2, c2)

totData(data = d2, r = 3, base = 2, nSpl = 15, nCTL = 2,
na.rm = TRUE)
```

Index