The analysis of rtPCR data

Jitao David Zhang, Markus Ruschhaupt

October 17, 2016

With the help of this document, an analysis of rtPCR data can be performed. For this, the user has to specify several parameters described below in the '.Rnw' version of this document. After that the software R can be used for the analysis by processing the '.Rnw' version. In the beginning you will find a short description of the calculated values. After that the parameters that have to be specified are explained.

1 The calculated values

Several values are calculated during the execution of this document. Some of them are explained here. The values explained here are calculated if no efficiencies are used (see below).

1. The mean or median of the replicates for one gene/sample combination is the Ct value.
2. For a sample A the Ct value of the housekeeping gene (or the median of the Ct values of all housekeeping genes) is subtracted from the corresponding Ct value of a gene $Gen1$. The result is the dCt value for $Gen1$ and sample A. This value is used for the t-test and the Wilcoxon test.
3. For a gene $Gen1$ the dCt value of the reference sample (or the mean of the dCt values of all reference samples) is subtracted from the corresponding dCt value of $Gen1$ and a sample B. This is called the ddCt value for $Gen1$ and sample B.
4. The transformation $x \to 2^{-x}$ is applied to each ddCt value. The resulting value is called 'level'.

Additionally, for each value an error is calculated. This is based on the standard error of the mean if the mean is used to summarize the triplets. If the median is used for summarization of the individual Ct values, the MAD divided by the square root of all samples is used as an error estimate.

2 Preparation

The following programs and packages have to be installed on the system.

- R (at least version 2.2.0)
- R package Biobase (at least version 1.7)
- R package RColorBrewer
- R package ddCt
- R package lattice
- R package xtable

In case of using this manuscript first time, you have to install the *ddCt* package. This process can be performed semi-automatically by copying the codes of the following section (titled *ddCtInstall*), which will download the current version of *ddCt* (dependent on the version of your R program) and install it.

```
> source("http://www.bioconductor.org/biocLite.R")
> biocLite(ddCt)
```

To run this document and perform the analysis you need a tab-delimited text file containing the individual Ct values of your experiment. This file can be exported from the software used to measure the Ct values. It is important that you export the individual Ct values (normally the software combines the replicates of the measurements to one value). After you have done this, you can follow this instruction step by step and set the appropriate parameters.

3 Choice of the parameters

First you have to specify the directory where your data is stored (load.path) and the directory where the results are supposed to be stored (save.path). You can specify the path relatively to the directory you are in when starting the script in R.

```
> load.path = "."
> save.path = "."
```

Then you have to specify the name of the file containing the individual Ct values (Ct.data.file.name). Additionally, you may specify a file containing additional data related to the samples (sample.annotation.file.name). This should be a tab-delimited text file with several columns and one row for each sample. Have a look at the example

'SampleAnno.txt'. The first column has to include the sample names and has to be named 'Sample'.

A file with additional information is important if you want to perform a t-test, if you use special colors for groups of samples, or if you want the samples to be grouped according to one column of the sample annotation file that can be specified through the parameter column.for.grouping. If all of this is not important to you, just set all parameters to 'NULL'. This also holds for other parameters described below that may be specifies but do not have to. If you do not want to specify such a parameter, just put 'NULL' there. If you have a sample annotation file and want to include only samples from this file into your final object, you can set the parameter 'useOnlySamplesFromAnno'.

```
> Ct.data.file.name       <-  c("")
> sample.annotation.file.name  <- NULL
> column.for.grouping          <- NULL
> useOnlySamplesFromAnno       <- FALSE
```

In case no input file was specified, a default file will be used for the demonstrative purpose.

Change gene names and sample names

You may change gene names and sample names, which is important for the final plot of your data. If you change gene/sample names you also change the ordering of the genes/samples in your plot.

In this example, 'Gen1' will become 'Gene A', 'Gen4' will become 'Gene B', and so on. In the final plot, Gene A will be plotted first, then Gene B followed by Gene C, HK1 and HK2. If you want to rename genes, **all** genes have to be included into this list. You cannot exclude genes that you do not want to be renamed. The sample names may be changed in the same way.

```
> #new.gene.names    <- c("Gen1"="Gene A",
> #                       "Gen4"="Gene B",
> #                       "Gen5"="Gene C",
> #                       "Gen2"="HK1",
> #                       "Gen3"="HK2")
> new.gene.names    <- NULL
> new.sample.names  <- NULL
```

Attention: If you rename genes or samples, the new names must be used for the parameters further down below.

Housekeeping genes and reference samples

Here you have to specify the housekeeping gene(s) and the reference sample(s). In this example, one reference sample and two housekeeping genes are used. If more than one object is specified, the names have to be given as shown in this example. If you specify more than one housekeeping gene, the algorithm will use the mean of the Ct values of the housekeeping genes for normalization. If you use more than one sample, the algorithm uses the mean of the chosen samples as the reference line.

```
> name.referenz.sample <- c("Sample2")
> name.referenz.gene   <- c("Gene3")
```

You may set a threshold for the Ct values. All individual Ct values values above this threshold will be treated as 'undetermined'.

```
> Threshold.for.Ct <- 40
```

Here you have to specify if you want to use the 'mean' or the 'median' for merging the individual Ct values for a gene/sample combination. The median is often more appropriate because it is more robust.

```
> TYPEOFCALCULATION <- "mean"
```

Efficiencies

You may include efficiencies for each gene. There is also the possibility to include error estimates for the efficiencies (for example a standard deviation). These estimates will be used for the error calculation. These efficiencies can be specified in the following way.

```
> EFFICIENCIES <- c("Gene A"=1.9,"Gene B"=1.8,"HK1"=2,"Gene C"=2,"HK2"=2)
> EFFICIENCIES.ERROR <- c("Gene A"=0.01,"Gene B"=0.1,"HK1"=0.05,"Gene C"=0.01,"HK2"=0
```

If you use efficiencies, only the raw Ct value and the final 'level' is calculated. There are no 'dCt' or 'ddCt' values. Hence no t-test can be performed if efficiencies are used. In this example we do not use efficiencies.

```
> EFFICIENCIES <- NULL
```

```
> EFFICIENCIES.ERROR <- NULL
```

Plot parameter

The following parameters are used to change the final plot. First you have to specify what you want to plot. Here you can specify any or both of the following two choices:

- level - For each gene and sample the relative expression to the reference line is plotted
- Ct - the raw, unnormalized but merged Ct values are plotted

```
> TheKindOfPlot <- c("level","Ct")
```

Sometimes genes and/or samples are not wanted to be included in the final plot. And sometimes you only want to include a small fraction of genes and/or samples. Have a look at the examples. We do not want the "NTC" sample to appear in the plot:

```
> #REMAIN
> names.of.the.genes.REMAIN.in.Output   <- NULL
> names.of.the.samples.REMAIN.in.Output <- NULL
> #NOT
> names.of.the.genes.NOT.in.Output      <- NULL
> names.of.the.samples.NOT.in.Output    <- NULL
```

Now you have to consider the following questions: Do you want the final plot to be drawn in a way such that the samples are plotted next to each other ?

```
> GROUPINGBYSAMPLES <- TRUE
```

And would you like each gene or sample to have its own plot?

```
> own.plot.for.each.object     <- TRUE
```

This is often useful if you have many genes or samples. Depending on the parameter GROUPINGBYSAMPLES either each gene (GROUPINGBYSAMPLES=TRUE) will have its own plot, or each sample (GROUPINGBYSAMPLES = FALSE) will have its own plot.

If you have a single plot for each individual gene, then you may color the sample bars according to one parameter of your sample annotation file (if you have specified such a file in the beginning of this script). You may also specify the colors.

```
> GroupingForPlot <- NULL
> GroupingColor   <- c("#E41A1C","#377EB8","#4DAF4A","#984EA3","#FF7F00")
```

You may specify a CUTTOFF for the y axis for all plots. Then all plots have the same scale.

```
> CUTOFF <- NULL
```

With the parameter BREWERCOL you can specify color sets you want to use in order to color the individual bars. For additional information have a look at the description of the 'RColorBrewer' package.

```
> BREWERCOL <- c("Set3","Set1","Accent","Dark2","Spectral","PuOr","BrBG")
```

Would you like to plot a legend? (TRUE/FALSE). Sometimes the plot will be messed up if a legend is plotted, so please have a try.

```
> LEGENDE <- TRUE
```

t-test and Wilcoxon test specification

You may perform a t-test and a Wilcoxon test if you have specified a sample annotation file above. **Attention:** A t-test and a Wilcoxon test will not work if efficiencies are used for the calculation.

```
> perform.ttest <- FALSE
```

If you want to do a t-test/Wilcoxon test, you have to specify the name of the column of your sample information file that includes the group information needed for the tests. If there are more than two groups in this column, tests for each possible pair of groups are performed.

```
> column.for.ttest <- NULL
```

If you want to perform a paired t-test/Wilcoxon test, you have to specify a column of your sample information file that contains information describing the pairing of the samples. A pair of samples must have the same number/parameter in that column. Please have a look at the example.

```
> column.for.pairs     <- NULL      ## ansonsten NULL
```

You can specify whether you want to exclude some samples from the t-test. Here we again want to exclude the 'NTC' sample.

```
> names.of.the.samples.REMAIN.in.ttest <- names.of.the.samples.REMAIN.in.Output
> names.of.the.samples.NOT.in.ttest    <- names.of.the.samples.NOT.in.Output
>
```

Housekeeping gene correlation

If two or more housekeeping genes are used, the correlation (Pearson and Spearman) for each pair of housekeeping genes is calculated. Additionally a plot is produced. You may specify some samples that are not supposed to be used for the correlation calculation, for example a no template control. In the default setting those samples are excluded that are also excluded from the plot.

```
> names.of.the.samples.REMAIN.in.cor <- names.of.the.samples.REMAIN.in.Output
> names.of.the.samples.NOT.in.cor    <- names.of.the.samples.NOT.in.Output
```

Now all parameters have been set. You have to start R, change the directory if you like, and then type the following:

Sweave("RT-PCR-Script-ddCt.Rnw")

4 Results

Various files are created during the calculation process. The most important files are the following:

- A tab-delimited text file containing all calculated values for each gene/sample combination, e.g. Ct+error, dCt+error, ddCt+error, level+error. **Name of the file:** 'allValues' followed by the name of the file containing the individual Ct values and extension.
- A .pdf file including the plots, either of the 'levels' or the raw Ct values (depending on your choice) **Name of the file:** 'Level' or 'Ct' followed by 'Result', the name of the file containing the individual Ct values and the extension. In this example the name is 'LevelResultTest.pdf'.
- A tab-delimited text file containing the level and the error of the level. This file can be used in Excel to create own plots with error bars. **Name of the file:** 'LevelPlusError' followed by the name of the file containing the individual Ct values and extension.
- A tab-delimited text file containing the results from the t-test and the Wilcoxon test (if these tests have been performed). **Name of the file:** 'ttest' followed by the name of the groups used in the test, followed by the name of the file containing the individual Ct values and extension. In this example we have: 'ttestG1G2Test.txt'.
- An R object containing all calculated values for each gene/sample combination, e.g. Ct+error, dCt+error, ddCt+error, level+error. Furthermore the object includes additional sample information. **Name of the file:** 'Result' followed by the name of the file containing the individual Ct values and extension.

There are additional tab-delimited text files and .html files containing 'Ct','dCt', 'ddCt' and 'level' information. All this information is also included in the 'allValues' file, but in a different format.
If warnings have been created during the calculation process they appear here:

Impressum

The following packages have been used for the calculation:

```
> toLatex(sessionInfo())
```

- R version 3.3.1 (2016-06-21), `x86_64-pc-linux-gnu`

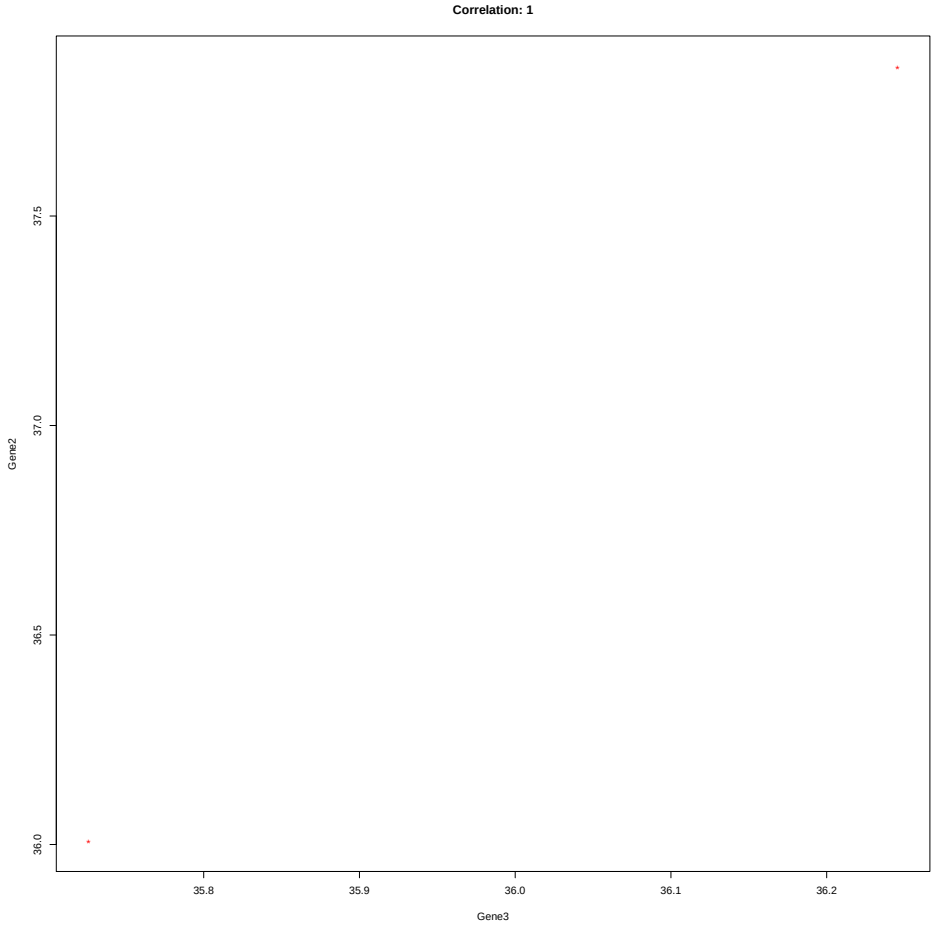

Figure 1: Correlation of the housekeeping genes

- Locale: LC_CTYPE=en_US.UTF-8, LC_NUMERIC=C, LC_TIME=en_US.UTF-8, LC_COLLATE=C, LC_MONETARY=en_US.UTF-8, LC_MESSAGES=en_US.UTF-8, LC_PAPER=en_US.UTF-8, LC_NAME=C, LC_ADDRESS=C, LC_TELEPHONE=C, LC_MEASUREMENT=en_US.UTF-8, LC_IDENTIFICATION=C
- Base packages: base, datasets, grDevices, graphics, methods, parallel, stats, utils
- Other packages: Biobase 2.34.0, BiocGenerics 0.20.0, RColorBrewer 1.1-2, ddCt 1.30.0, lattice 0.20-34
- Loaded via a namespace (and not attached): grid 3.3.1, tools 3.3.1, xtable 1.8-2

Change Logs

Version 4.0

- Add change logs to the end of the file
- Add errBarplot for both detector and sample views
- Adjust to the new ddCt package (developed by Jitao David Zhang and Rudolf Biczok) which was submitted to the Bioconductor community