R as an Environment for the Analysis of dPCR and qPCR Experiments

by Stefan Rödiger, Michał Burdukiewicz, Konstantin Blagodatskikh and Peter Schierack

Abstract There is an ever-increasing number of applications, which use quantitative PCR (qPCR) or digital PCR (dPCR) to elicit fundamentals of biological processes. Moreover, the novel amplification strategies based on quantitative isothermal amplification (qIA) have become more prominent in life sciences and point-of-care-diagnostics. Additionally, the analysis of melting data is essential during many experiments. Several software packages have been developed for the analysis of such data sets. In most cases, the software is either distributed as closed source software or as monolithic block with little freedom to perform highly customized analysis procedures. Others and we argue that R is an excellent foundation for reproducible and transparent data analysis in a highly customizable cross-platform environment. However, for novices it is often challenging to master R or learn capabilities of the vast number of packages available. In the paper, we describe exemplary workflows for the analysis of qPCR, qIA or dPCR experiments including the analysis of melting curve data. Our analysis relies entirely on R packages available from public repositories. Additionally, we provide information related to standardized and reproducible research with R.

Introduction

The quantitative Polymerase Chain Reaction (qPCR) is the method of choice when a precise quantification of minute DNA traces is required. The applications include the detection and quantification of pathogens or gene expression analysis (Peirson et al., 2003). Only few bioanalytical applications had such a significant impact on the progress of life sciences and medical sciences as the qPCR. Numerous commercial and experimental monitoring platforms have been developed in the past years. This includes standard plate cyclers, capillary cyclers, microfluidic platforms and related technologies (Rödiger et al., 2013b; Devonshire et al., 2013; Viturro et al., 2014; Rödiger et al., 2014; Khodakov and Ellis, 2014; Wu et al., 2014).

In the past decades several isothermal amplification technologies emerged, such as helicase dependent amplification (HDA), as alternative to PCR. The isothermal amplification was readily combined with real-time monitoring technologies (qIA) and is used nowadays in various fields like diagnostics and point-of-care-testing (Rödiger et al., 2014).

The digital PCR (dpcR) is a novel approach for detection and quantification of nucleic acids and can be seen as a next generation method. The dPCR technology breaks fundamentally with the previous concept of nucleic acid quantification. The key difference between dPCR and traditional qPCR lies in the method of measuring (absolute) nucleic acids amounts, which yields discrete information instead of the continuous signal. This is possible after "clonal DNA amplification" in thousands of small separated partitions (e.g., droplets, nano chambers; Huggett et al. 2013; Milbury et al. 2014; Morley 2014). The partitions with no nucleic acid remain negative and the others turn positive (e.g., Figure 1). Selected technologies (e.g., OpenArray®Real-Time PCR System) monitor amplification reactions in the chambers ("partition") in real-time. After that, all Cq values are calculated from the amplification curves and converted into discrete events by means of positive and negative chambers. Finally, the absolute quantification of nucleic acids is done using Poisson statistics. Recently, we have published the **dpcR** package at CRAN, which is the first open source R software package for the analysis of digital PCR (dPCR) experiments (see package **dpcR** for details).

Many of the commercial and experimental hardware platforms provide means to analyse the data. The complexity of hardware, wetware and software requires expertise to master a workflow. This comprises standards for experiment design, generation and analysis of data, interpretation, reporting and storage of results (Huggett et al., 2014). The scientific misconduct and fraud have shaken the scientific community on several occasions (Fang et al., 2012). In particular, the scientific community works hard to uncover pitfalls of qPCR experiments. This lead to the development of peer-reviewed quantification cycle (Cq) analysis algorithms (Ruijter et al., 2013), throughout characterized qPCR chemistries (Ruijter et al., 2014) and guidelines for a proper conduct of qPCR experiments as implemented in the MIQE guidelines (minimum information for publication of quantitative real-time PCR experiments; Bustin et al.; Huggett et al. 2013). We share the philosophy of the MIQE guidelines to increase the experimental transparency for better experimental practice and reliable interpretation of results and encourage the use of open data exchange formats like the XML-based Real-Time PCR Data Markup Language (RDML; Lefever et al. 2009). We see the application of R in line with this philosophy.

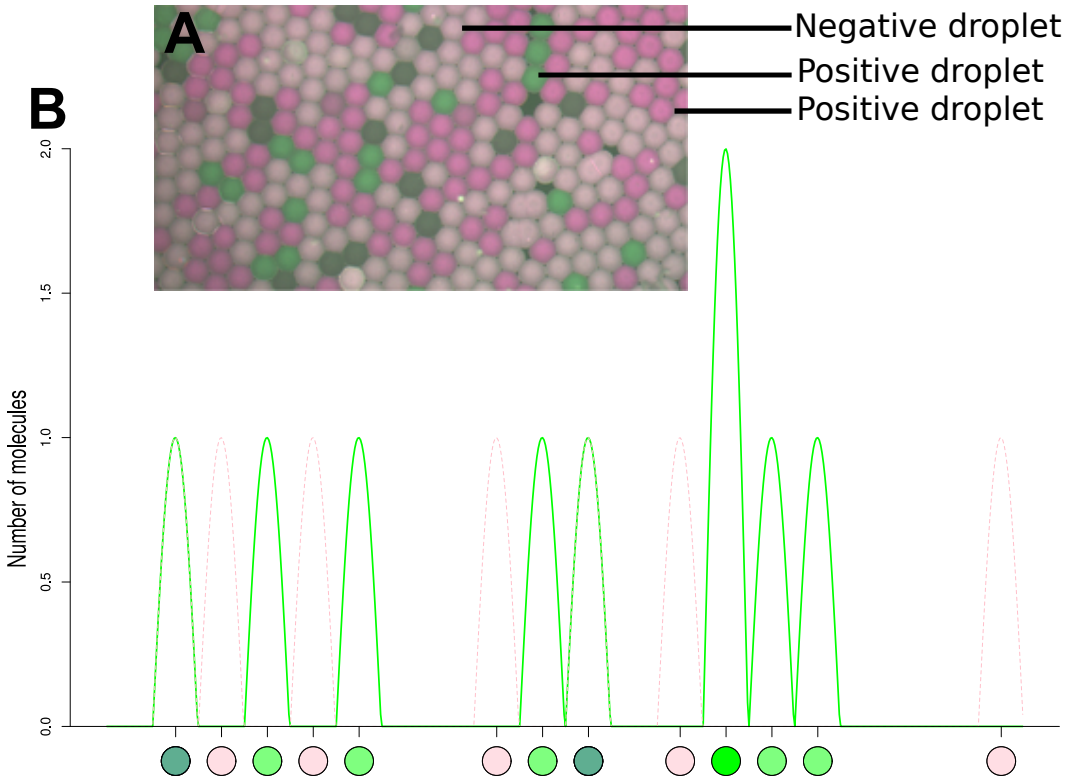

Figure 1: Scheme of droplet digital PCR experiment. **A)** A droplet digital PCR reaction mix was formed in a Bio-Rad QX100 system. The droplets (circa 100 μm in diameter) were subjected to custom made slide chambers for the detection and analysis in fully automatized imaging VideoScan platform (Rödiger et al., 2013b). **(B)** Subsequently the samples can be digitalized by counting number of positive and total number of droplets. The plot was generated with the `sim_ddpcr` function from the **dpcR** package.

In case of closed source software usually the analysis happens in a black box fashion tied to a specific platform. We agree that black boxes are not necessarily a bad thing, but should be avoided if possible (Rödiger et al., 2013a; Spiess et al., 2015). Studies by McCullough and Heiser (2008); Almiron et al. (2010); Durán et al. (2014) exemplified where the black box approach might fail in science. The same holds true for the open source software since software bugs are independent of a development model. However, at least open source gives the possibility to track and eliminate errors by an individual entity. Often black box systems have limitations in the data processing and force the user to process the data by suboptimal analysis algorithms (Ruijter et al., 2013). The visualization options are usually limited by the software vendor preferences and do not attain publication quality. Users who have no access to the commercial software are barred. Aside from closed source software, data analysis is often performed in spreadsheets. However, this data processing approach is not advisable for research purposes. Most spreadsheets lack (or do not use) tools to validate the input, to debug implemented procedures and to automatize the workflow. These traits make them prone to errors and not well suited for complicated tasks (McCullough and Heiser, 2008; Burns, 2014).

For several reasons, R is one of the most popular tools in bioinformatics and is known as an early adopter of emerging technologies (Pabinger et al., 2014). R provides packages to build highly customized workflows, covering: data read-in, data preprocessing, analysis, post-processing, visualization and storage. As recently briefly reviewed in Pabinger et al. (2014), numerous R packages have been developed for the analysis of qPCR, dPCR, qIA and melting curve analysis experiments, including: **kulife** (Ekstrom et al., 2013), **MCMC.qpcr** (Matz, 2015), **qPCR.CT** (Pan et al., 2012), **DivMelt** (Swan, 2013), **qpcR** (Spiess, 2014), **dpcR** (Burdukiewicz et al., 2015), **chipPCR** (Roediger and Burdukiewicz, 2014), **MBmca** (Roediger, 2015), **RDML** (Blagodatskikh et al., 2015), **nondetects** (McCall1 et al., 2014), **qpcrNorm** (Mar, 2009), **HTqPCR** (Dvinge and Bertone, 2009), **SLqPCR** (Kohl, 2007), **ddCt** (Zhang et al., 2015), **EasyqpcR** (Pape, 2012), **unifiedWMWqPCR** (Neve et al., 2014; Neve and Meys, 2014), **ReadqPCR** and **NormqPCR** (Perkins et al., 2012). All the packages are either available from CRAN or Bioconductor (Gentleman et al., 2004). The packages can be freely combined in a plugin-like architecture. R is an independent, cross-platform instrument and provides a broad spectrum of calculation options. Particularly, the visualization of experiments is one of R's pinnacles. R enables the users to

create an efficient manipulation, restructuring and reshaping of data to make them readily-available for further processing. This is of particular importance to the human-machine interface (Oh, 2014). The intrinsic properties of R such as the naming convention (Bååth, 2012) and class systems (e.g., **S3**, **S4**, reference classes and **R6**) vary considerable, depending on the package developer preferences. However, due to the open source approach, there is the common ground to track numerical errors. R offers various methods for a standardized data import/export and exchange. Workflows can embed structured models (Zeller et al., 2009), open data exchange formats (e.g., RDML), binary formats (Michna and Woods, 2013) or tools provided by the R workspace (R Core Team, 2015). Additionally, the R environment offers several data sets, which can be used for testing of algorithms. Therefore, others and we argue that R is suitable for reproducible research (Gesmann and de Castillo, 2011; Murrell, 2012; Gandrud, 2013; Hofmann et al., 2013; Leeper, 2014; Liu and Pounds, 2014).

The aim of this paper is to show case studies for qPCR, dPCR, qIA and melting curve analysis experiments. Our workflow effectively follows the principle illustrated in Figure 2. We intent to aggregate functionalities dispersed between various packages and offer a fast insight in the analysis of nucleic acid experiments with R. In particular, we describe how to:

- read-in data from a standardized file format,
- pre-process the amplification curve data,
- calculate specific parameters from the amplification curve data,
- calculate the melting temperature,
- and report the data.

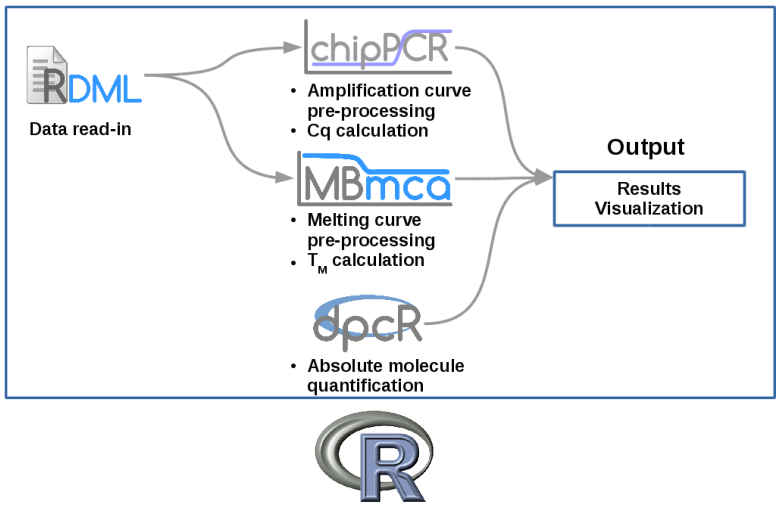

Figure 2: Exemplary workflow for qPCR, dPCR, qIA and MCA experiments in R. Core functionality is provided by the R software environment for statistical computing and graphics. In our scenario we used the **RDML** package to read-in data in standardized format. However, any format supported by R can be used. Further, processing of amplification curve data was performed with the **chipPCR** package and melting curve data were analysed with the **MBmca** package. The **dpcR** package can be embedded in the analysis of digital PCR experiments. Cq, quantification cycle; T_M, melting temperature.

Setting-up a working environment

We recommend performing the scripting in a dedicated integrated development environment (IDE) and graphical user interface (GUI) such as RKWard (Rödiger et al., 2012), Rstudio[1] or related technologies (Valero-Mora and Ledesma, 2012). Benefits of IDE's with GUI include syntax-highlighting, auto completion and function references for rapid prototyping of workflows.

Typically, the analysis will start with data from a commercial platform. Most platforms have an option to export a CSV file or spreadsheets application file (e.g., *.xls, *.odt). The details for the data import have been described elsewhere (R Core Team, 2015; Rödiger et al., 2012). To keep the case study sections compact we have chosen to load datasets from the **qcpR** package (Ritz and Spiess, 2008; Spiess et al., 2008) (v. 1.4.0) and the **RDML** package (v. 0.4-2) to our workspace. The **chipPCR** package

[1] http://www.rstudio.com/

(v. 0.0.8-3) was used for data preprocessing, quality control and the calculation of the quantification cycle (Cq). The Cq is a quantitative measure, which represents the number of cycles needed to reach a user defined threshold signal level, in the exponential phase of a qPCR/qIA reaction. Several Cq methods have been described (Ruijter et al., 2013). In this study we have chosen the second derivative maximum method (Cq_{SDM}) and the "Cycle threshold" (Cq_{Ct}) method.

During a perfect qPCR reaction, the target DNA doubles (2^n; n = cycle number) at each cycle. Here the amplification efficiency (AE) is 100 %. However, in reality, numerous factors cause an inhibition of the amplification ($AE < 100$ %). The AE can be determined by the relation of the Cq value depending on the sample input quantity as described in Roediger and Burdukiewicz (2014).

In Rödiger et al. (2013a) we described the application of R for the analysis of melting curve experiments from microbead-based assays. Since the mathematical foundation for melting curve analysis (MCA) is identical between all platforms we applied the functions from the **MBmca** package (0.0.3-4) for an analysis of the target specific melting temperature (T_M) in experiments of the present study.

We completed our examples with a case study for the analysis of dPCR experiment. In particular, we used the **dpcR** (0.1.3.1) to estimate the number of molecules in a sample.

Results

In the following sections we will show how R can be used (I) as an unified open software for data analysis and presentation in research, (II) as software frame-work for novel technical developments, (III) as platform for teaching of new technologies and (IV) as reference for statistical methods.

Case study one – qPCR and amplification efficiency calculation

The goal of our first case study was to calculate the Cq values and the AE from a qPCR experiment. We used the "guescini1" dataset from the **qcpR** package, where the gene *NADH dehydrogenase 1* (MT-ND1) was amplified in a LightCycler® 480 (Roche) thermocycler. Details of the experiment are described in Guescini et al. (2008). First we started with loading the required packages and datasets. A good practice for reproducible research is to track the package versions and environment used during the analysis. The function `sessionInfo()` from the **utils** package provides this information. Assuming that the analysis starts with a clean R session it is possible to assign the required packages to an object, as shown below. The reproducibility of research can be further improved by the **archivist** package (Biecek and Kosinski, 2015), which stores and recovers crucial data and preserves metadata of saved objects (not shown). All settings of an R session can be easily saved and/or restored using the **settings** package (van der Loo, 2015).

```
# Load the required packages for the data import and analysis.
# Load the chipPCR package for the pre-processing and curve data quality
# analysis and load the qpcR package as data resource.
require(chipPCR)
require(qpcR)

# Collect information about the R session used for the analysis of the
# experiment.
current.session <- sessionInfo()

# Next load the 'guescini1' dataset from the qpcR package to the
# workspace and assign it to the object 'gue'.
gue <- guescini1

# Define the dilution of the sample DNA quantity for the calibration curve
# and assign it to the object 'dil'.
dil <- 10^(2:-4)
```

We previewed the amplification curve raw data using the `matplot` function (see code below). The amplification curve data showed a strong signal level variation in the plateau region (Figure 3A). Therefore, all data were subjected to a minimum-maximum normalization (compare Rödiger et al. 2013a) using the `CPP` function from the **chipPCR** package. In addition, all data were baselined and smoothed (Figure 3B). The Cq values were calculated by the Cq_{SDM} and Cq_{Ct} methods as shown next.

```
# Pre-process the amplification curve data with the CPP function from the
# chipPCR package. The trans parameter was set TRUE to perform a baselining and
```

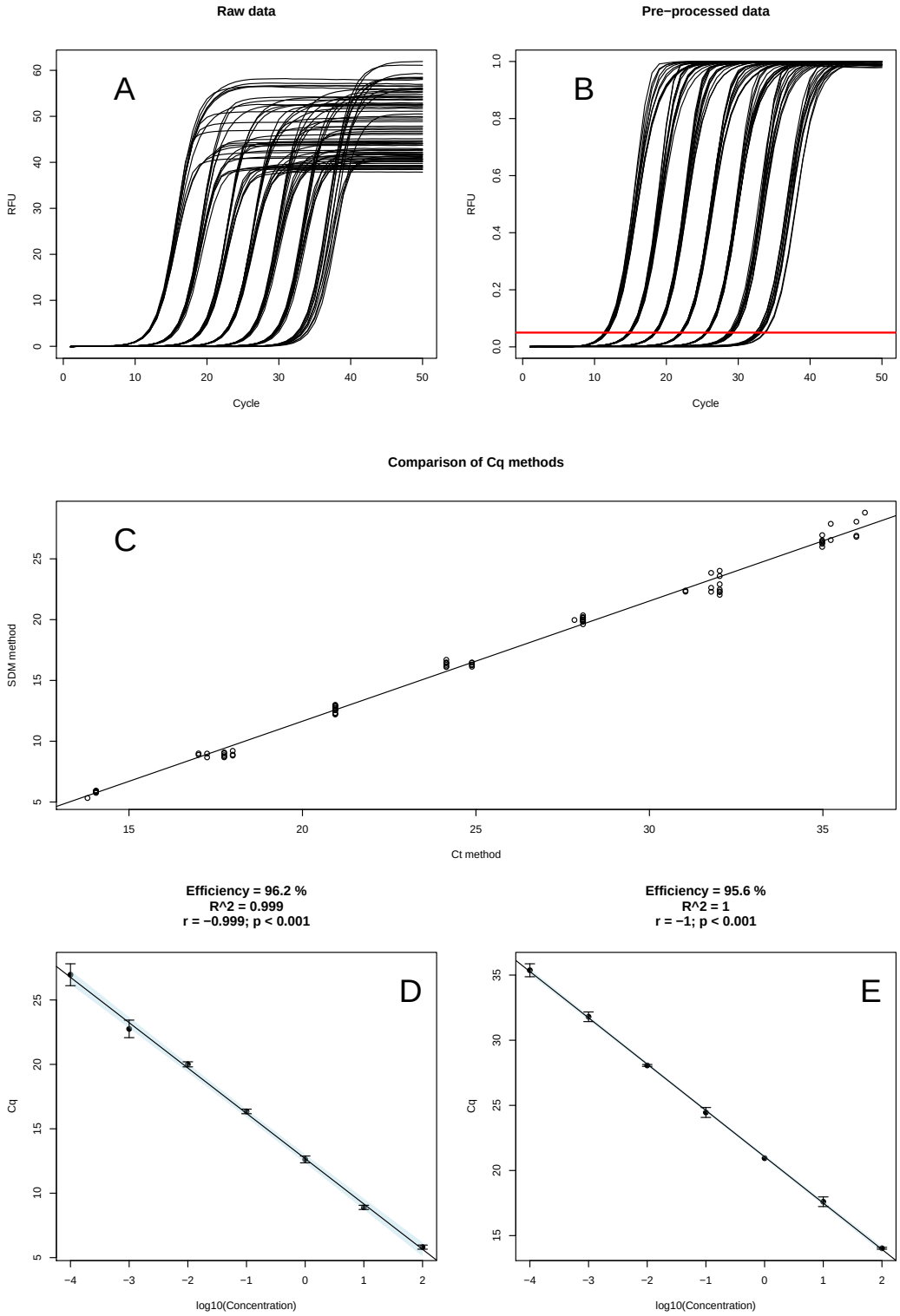

Figure 3: Analysis of the amplification curve data of the `guescini1` dataset. **(A)** Raw data from the calibration curve samples were visually inspected. The qPCR curves display a broad variation in plateau fluorescence (38–62 RFU). The red horizontal line indicates the fluorescence level (0.05) used for the calculation of the Cq value by the "cycle threshold" method. **(B)** The `CPP` function from the **chipPCR** package was used to baseline the data, to smooth the data with Savitzky-Golay smoothing filter and to normalize the data between 0 and 1. **(C)** The Cq values were calculated by the Cq_{SDM} method ("SDM method") (`inder`, **chipPCR**) and the Cq_{Ct} method ("Ct method") (`th.cyc`, **chipPCR**). The threshold value was set to $r = 0.05$. The Cq_{SDM} and Cq_{Ct} values were plotted and analysed by a linear regression ($R^2 = 0.9945$; $P < 2.2^{-16}$) and Pearson's ($r = 0.9972605$; $P < 2.2^{-16}$). The AE based on **(D)** Cq_{Ct} values and **(E)** Cq_{SDM} values were automatically analysed with the `effcalc` (**chipPCR**) function.

```
# the method.norm parameter was set to minm for a min-maximum normalization. All
# amplification curves were smoothed by Savitzky-Golay smoothing.

res.CPP <- cbind(gue[, 1], apply(gue[, -1], 2, function(x) {
  CPP(gue[, 1], x, trans = TRUE, method.norm = "minm",
      bg.range = c(1, 7))[["y.norm"]]
}))

# Use the th.cyc function from the chipPCR package to calculate the Cq values
# by the cycle threshold method at a threshold signal level "r" of 0.05.
Cq.Ct <- apply(gue[, -1], 2, function(x)
  th.cyc(res.CPP[, 1], x, r = 0.05)[1])

# Use the inder function from the chipPCR package to calculate the Cq values
# by the SDM method.
Cq.SDM <- apply(gue[, -1], 2, function(x)
  summary(inder(res.CPP[, 1], x))[2])

# Fit a linear model to carry out a regression analysis.
res.Cq <- lm(Cq.Ct ~ Cq.SDM)
```

To compare the Cq_{SDM} and Cq_{Ct} methods we performed a regression analysis. The Cq methods are in a good agreement. However, the dispersion of the Cq_{Ct} values appeared to be higher than in the Cq_{SDM} values (Figure 3C).

```
> summary(res.Cq)

Call:
lm(formula = Cq.Ct ~ Cq.SDM)

Residuals:
    Min      1Q  Median      3Q     Max
-1.4904 -0.2730  0.0601  0.3540  1.1871

Coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept) -8.125534   0.207419  -39.17   <2e-16 ***
Cq.SDM       0.988504   0.008097  122.08   <2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.5281 on 82 degrees of freedom
Multiple R-squared:  0.9945,  Adjusted R-squared:  0.9945
F-statistic: 1.49e+04 on 1 and 82 DF,  p-value: < 2.2e-16
```

The dilution ("dil") and the Cq ("Cq.Ct") values served as input for the calculation of the amplification efficiency (AE) with the `effcalc` function from the **chipPCR** package. In our case study we needed to rearrange to "Cq.Ct" values in a matrix using the command `effcalc(dil,t(matrix(Cq.Ct,nrow = 12,ncol = 7))`. For visualization of the confidence intervals of the regression analysis we set the parameter `CI = TRUE`. Finally, Cq values were plotted using the `layout` function (Figure 3D and E).

```
layout(matrix(c(1, 2, 3, 3, 4, 5), 3, 2, byrow = TRUE))
# Set bigger top margin.
par(mar = c(5.1, 4.1, 6.1, 2.1))

matplot(gue[, -1], type = "l", lty = 1, col = 1, xlab = "Cycle",
        ylab = "RFU", main = "Raw data")
legend("topleft", "A", cex = 3, bty = "n")

matplot(res.CPP[, -1], type = "l", lty = 1, col = 1, xlab = "Cycle",
        ylab = "RFU", main = "Pre-processed data")
legend("topleft", "B", cex = 3, bty = "n")
abline(h = 0.05, col = "red", lwd = 2)

plot(Cq.SDM, Cq.Ct, xlab = "Ct method", ylab = "SDM method",
```

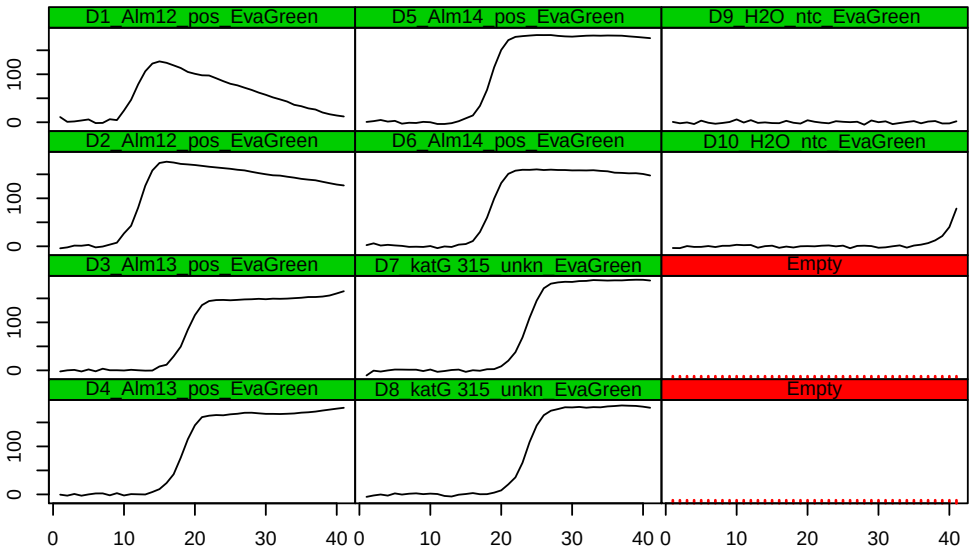

Figure 4: Analysis of the amplification curve data. The calibration curve samples were inspected by the `plotCurves` function from the **chipPCR** package. The green color code indicates that the data contain no missing values. However, the visual inspection revealed that the data are noisy. All samples ("D1_Alm12…"–"D8_Alm12…") appeared to be positive. One negative control ("D10_H20_ntc_EvaGreen") seems to be contaminated.

```
     main = "Comparison of Cq methods")
abline(res.Cq)
legend("topleft", "C", cex = 3, bty = "n")

plot(effcalc(dil, t(matrix(Cq.Ct, nrow = 12, ncol = 7))), CI = TRUE)
legend("topright", "D", cex = 3, bty = "n")

plot(effcalc(dil, t(matrix(Cq.SDM, nrow = 12, ncol = 7))), CI = TRUE)
legend("topright", "E", cex = 3, bty = "n")

# Set top margin to default value.
par(mar = c(5.1, 4.1, 4.1, 2.1))
```

As shown in this case study, it is easy to set-up a streamlined workflow for data read-in, pre-processing and analysis with a few functions.

Case study two – qPCR and melting curve analysis

A common task during the analysis of qPCR experiments is to distinguish between positive and negative samples (compare Figure 4). If the melting temperature of a sample is known it is possible to automatize the decision by a melting curve analysis (MCA). As shown in Rödiger et al. (2013a) this can be done by interrogating the T_M. Therefore, we used a logical statement, which tests if T_M is within a tight temperature range. We used the signal height as second parameter. In line with "Case study one" we used the function `sessionInfo()` to track all packages used during the analysis. Reproducible research is greatly enhanced if open data exchange formats are used. Therefore, we used the **RDML** package for data read-in. The amplification and melting curve data were measured with a CFX96 system (Bio-Rad) and then exported as RDM v 1.1 format file as 'BioRad_qPCR_melt.rdml'. Within this qPCR experiment we amplified the *Mycobacterium tuberculosis katG*[2] gene and tried to detect a mutation at codon 315. The experiment was separated in two parts:

1. Detection of overall *M. tuberculosis* DNA (wild-type and mutant) and
2. specific detection of wild-type *M. tuberculosis* by melting of TaqMan probe (quencher – BHQ2, fluorescent reporter – Cy5) with amplified DNA (see Luo et al. 2011 for probe/primer sequences and further details).

[2] https://www.wikigenes.org/e/gene/e/923602.html

The qPCR was conducted using EvaGreen® Master Mix (Syntol) according to the manufacturer's instructions, with 500 nM of primers and probe in a 25 μL final reaction volume. Thermocycling was conducted using a CFX96 (BioRad) initiated by 3 min incubation at 95 °C, followed by 41 cycles (15 s at 95 °C; 40 s at 65 °C) with a single read-out taken at the end of each cycle. Probe melting was conducted between 35 °C and 95 °C by 1 °C at 1 s steps.

The structure of an RDML file is quite complex. A simple and fast method to compactly display the structure of such an object is to use the `str` function (not shown). Only a subset of the data was used in our case study and combined to the object "qPCR".

```
# Import the qPCR and melting curve data via the RDML package.
# Load the chipPCR package for the pre-processing and curve data quality
# analysis and the MBmca package for the melting curve analysis.
require(RDML)
require(chipPCR)
require(MBmca)

# Collect information about the R session used for the analysis of the qPCR
# experiment.
current.session <- sessionInfo()

# Load the BioRad_qPCR_melt.rdml file form RDML package and assign the data to
# the object BioRad.

filename <- paste(path.package("RDML"), "/extdata/",
                  "BioRad_qPCR_melt.rdml", sep = "")
BioRad <- RDML(filename, name.pattern = "%TUBE%_%NAME%_%TYPE%_%TARGET%")

# Fetch cycle dependent fluorescence for the EvaGreen channel of the
# katG gene and aggregate the data in the object qPCR.

qPCR <- cbind(BioRad[["qPCR"]][["EvaGreen"]][["pos"]],
              BioRad[["qPCR"]][["EvaGreen"]][["unkn"]][, -1],
              BioRad[["qPCR"]][["EvaGreen"]][["ntc"]][, -1])

# Leave data only from row 'D' that contains target 'Cy5-2' at channel 'Cy5'
qPCR <- cbind(qPCR[,1], qPCR[, grep("^D", names(qPCR))])
```

We inspected and pre-processed a subset of the amplification curve data solely using functionalities provided by the **chipPCR** package. The `plotCurves` function was used to get an overview of the curvatures. The data indicated a baseline shift in all curves with a slight negative trend (Figure 4). This observation suggested to baseline the raw data by using a linear regression model (cycles x – y; ($bg.range = c(x, y)$) in the `CPP` function.).

```
# Use plotCurves function to get an overview of the amplification curve samples.

plotCurves(qPCR[, 1], qPCR[, -1], type = "l")

# Detect positive samples - calculate Cq values by the cycle threshold method.
# The threshold signal level r was set to 10.
Cq.Positive <- t(apply(qPCR[, -1], 2, function(x) {
  res <- CPP(qPCR[, 1], x, trans = TRUE, bg.range = c(2, 8))[["y.norm"]]
  th.cyc <- th.cyc(qPCR[, 1], res, r = 10)[1]
  cq <- as.numeric(th.cyc)
  pos <- !is.na(cq)
  c(Cq = cq, M.Tub_positive = pos)
}))
Cq.Positive
```

Since the amplification curves indicated that selected samples (except non-template-control ("NTC")) are positive, we distinguished between true positive and true negative samples by MCA (Figure 5A).

```
# Fetch temperature dependent fluorescence for the Cy5 channel of the
# probe that can hybridize with Mycobacterium tuberculosis katG gene (codon 315)
# and aggregate the data in the object 'melt'.
```

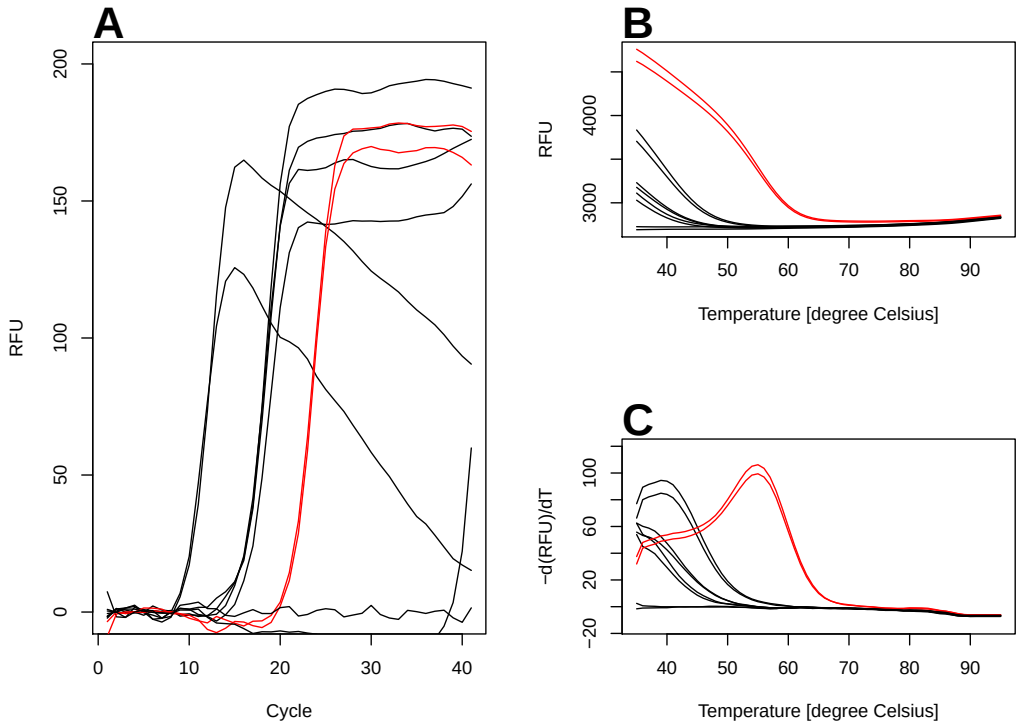

Figure 5: Graphical presentation of the amplification curve data and melting curve data. **(A)** The raw amplification curve data were pre-processed with the `CPP` function prior to visualization. To calculate the T_M values from the raw melting curve data **(B)** we used the `diffQ` function from the **MBmca** package. **(C)** We adjusted our algorithm to plot the true positive melting peaks in red, while negative melting peaks were labelled in black.

```
melt <- cbind(BioRad[["Melt"]][["Cy5-2"]][["pos"]],
              BioRad[["Melt"]][["Cy5-2"]][["unkn"]][, -1],
              BioRad[["Melt"]][["Cy5-2"]][["ntc"]][, -1])

# Calculate the melting temperature with the diffQ function from the MBmca
# package. Use simple logical conditions to find out if a positive sample with
# the expected Tm of circa 54.5 degree Celsius is found. The result of the test
# is assigned to the object 'positive'.
Tm.Positive <- matrix(nrow = length(melt[, -1]),
                      byrow = TRUE,
                      dimnames = list(names(melt)[-1]),
                      unlist(apply(melt[, -1], 2, function(x) {
                        res.Tm <- diffQ(cbind(melt[, 1], x),
                                        fct = max, inder = TRUE)
                        positive <- ifelse(res.Tm[1] > 54 &
                                           res.Tm[1] < 55 &
                                           res.Tm[2] > 80, 1, 0)
                        c(res.Tm[1], res.Tm[2], positive)
                      })))

# Present the results in a tabular output as data.frame 'results.tab'.
# Result of analysis logic is:
# Cq.Positive && Tm.Positive = Wild-type
# Cq.Positive && !Tm.Positive = Mutant
# !Cq.Positive && !Tm.Positive = NTC
# !Cq.Positive && Tm.Positive = Error
results <- sapply(1:length(Cq.Positive[,1]), function(i) {
  if(Cq.Positive[i, 2] == 1 && Tm.Positive[i, 3] == 1)
    return("Wild-type")
```

```
  if(Cq.Positive[i, 2] == 1 && Tm.Positive[i, 3] == 0)
    return("Mutant")
  if(Cq.Positive[i, 2] == 0 && Tm.Positive[i, 3] == 0)
    return("NTC")
  if(Cq.Positive[i, 2] == 0 && Tm.Positive[i, 3] == 1)
    return("Error")
})

results.tab <- data.frame(cbind(Cq.Positive, Tm.Positive, results))
names(results.tab) <- c("Cq", "M.Tub DNA", "Tm", "Height",
                        "Tm positive", "Result")

results.tab[["M.Tub DNA"]] <- factor(results.tab[["M.Tub DNA"]],
                                     labels = c("Not Detected", "Detected"))

results.tab[["Tm positive"]] <- factor(results.tab[["Tm positive"]],
                                       labels = c(TRUE, FALSE))
results.tab
```

The results of the analysis can be invoked by the statement `results.tab` (not shown). Finally, we plotted and printed the output of our analysis (Figure 5C).

```
# Convert the decision from the results.tab object in a color code:
# Negative, black; Positive, red.

color <- c(Tm.Positive[, 3] + 1)

# Arrange the results of the calculations in plot.
layout(matrix(c(1, 2, 1, 3), 2, 2, byrow = TRUE))

# Use the CPP function to preporcess the amplification curve data.
plot(NA, NA, xlim = c(1, 41), ylim = c(0,200), xlab = "Cycle", ylab = "RFU")
mtext("A", cex = 2, side = 3, adj = 0, font = 2)
lapply(2L:ncol(qPCR), function(i)
  lines(qPCR[, 1],
        CPP(qPCR[, 1], qPCR[, i], trans = TRUE,
            bg.range = c(1,9))[["y.norm"]],
        col = color[i - 1]))

matplot(melt[, 1], melt[, -1], type = "l", col = color,
        lty = 1, xlab = "Temperature [degree Celsius]", ylab = "RFU")
mtext("B", cex = 2, side = 3, adj = 0, font = 2)

plot(NA, NA, xlim = c(35, 95), ylim = c(-15, 120),
     xlab = "Temperature [degree Celsius]", ylab = "-d(RFU)/dT")
mtext("C", cex = 2, side = 3, adj = 0, font = 2)

lapply(2L:ncol(melt), function(i)
  lines(diffQ(cbind(melt[, 1], melt[, i]), verbose = TRUE,
              fct = max, inder = TRUE)[["xy"]], col = color[i - 1]))
```

Case study three – Isothermal amplification

Isothermal amplification is an alternative to PCR, which uses a constant temperature rather than cycling through denaturation, annealing and extension steps (Rödiger et al., 2014). The corresponding signal is monitored continuously on a time basis. Often the abscissa values are not uniformly spaced[3]. In qIA the Cq values are dependent on the time instead of cycles. In this study we used the `th.cyc` function to determine the time (Cq_t) required to reach a certain threshold signal level.

We performed a quantitative isothermal amplification (qIA) with the plasmid *pCNG1* by using a Helicase Dependent Amplification (HDA). Our previously reported VideoScan platform (Rödiger et al., 2013b) was used to control the temperature and to monitor the amplification reaction. The

[3]The `C81` example data set chosen in this case study are not uniformly spaced. The `CPP` function gives a warning in such a case.

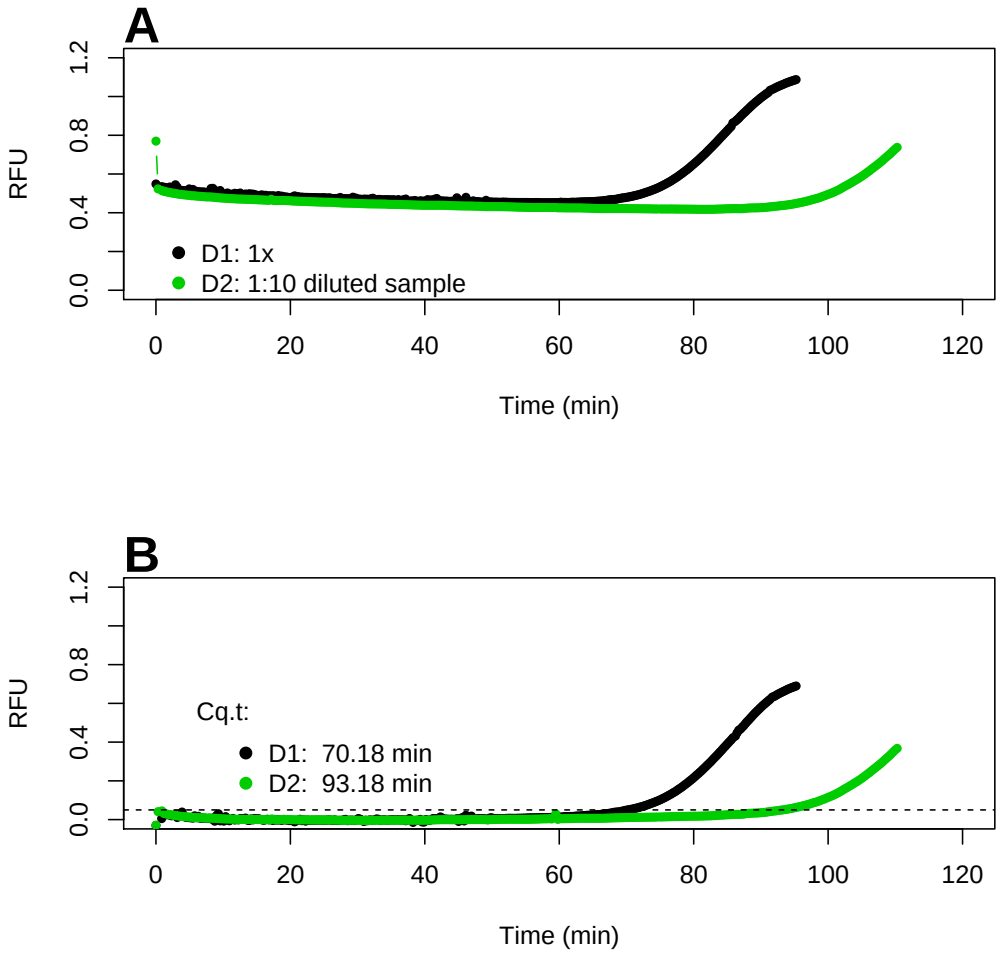

Figure 6: Quantitative isothermal amplification by Helicase Dependent Amplification (HDA). **(A)** The raw data of the HDA (D1, undiluted, D2 1 : 10 diluted) exhibit some outliers (detector artifacts), an off-set of circa 0.5 relative fluorescence units (RFU) and a slight negative trend in the baseline region (0–52 minutes). **(B)** First we used the `CPP` function to smooth the data with a spline function. Baselining was done with a linear regression model (robust MM-estimator). Finally, we used the `th.cyc` function to calculate the cycle threshold time for samples D1 and D2. The threshold value was set to $r = 0.05$ $(--)$.

VideoScan technology is based on a highly versatile fluorescence microscope imaging platform, which can be operated with a heating/cooling unit (HCU) for qPCR and MCA applications (Rödiger et al., 2013a,b). Since the enzyme DNA Helicase unwinds DNA, no thermal denaturation is needed. The HDA conditions were taken from the "IsoAmp III Universal tHDA Kit", Biohelix Corp, as described by the vendor. In detail, the reaction was composed of "mix A)" 10 μL A. bidest, 1.25 μL 10xb uffer, 0.75 μL primer (150 nM final), 0.5 μL template plasmid. Preincubation: The mixture was incubated for 2 min at 95°C and immediately placed on ice. Reaction "mix B)" contained 5 μL A. bidest., 1.25 μL 10x buffer, 2 μL NaCl, 1.25 μL $MgSO_4$, 1.75 μL dNTPs, 0.25 μL EvaGreen (Biotium), 1 μL enzyme mix. The mix was covered with 50 μL mineral oil (Roth). The fluorescence measurement in VideoScan HCU started directly after adding "mix B)" at 65°C. A $1x$ (D1) and a 1 : 10 dilution (D2) were tested. The resulting dataset `C81` is part of the **chipPCR** package. Two concentrations (stock and 1:10 diluted stock) of input DNA were used in the HDA. Similar to the previous case studies, we first prepared the plot of the data. Since the raw data showed a slight negative trend and an off-set of circa 0.45 RFU it was necessary to pre-process the raw data (Figure 6A).

```
# Drawn in an 2-by-1 array on the device by two columns and one row.
par(mfrow = c(2, 1))

# Plot the raw data from the C81 dataset to the first array and add
# a legend. Note: The abscissa values (time in seconds) was divided
# by 60 (C81[, i] / 60) to convert to minutes.
```

```
plot(NA, NA, xlim = c(0, 120), ylim = c(0, 1.2), xlab = "Time (min)", ylab = "RFU")
mtext("A", cex = 2, side = 3, adj = 0, font = 2)
lapply(c(2, 4), function(i) {
  lines(C81[, i] / 60, C81[, i + 1], type = "b", pch = 20, col = i - 1)
})
legend(10, 0.8, c("D1: 1x", "D2: 1:10 diluted sample"), pch = 19, col = c(1, 3),
       bty = "n")

# Prepare a plot on the second array for the pre-processed data.
plot(NA, NA, xlim = c(0, 120), ylim = c(0, 1.2), xlab = "Time (min)", ylab = "RFU")
mtext("B", cex = 2, side = 3, adj = 0, font = 2)
```

First, we used the CPP function to pre-process the raw data. Similar to the other case studies we baselined and smoothed the amplification curve data prior to the analysis of the Cq_t value. However, instead of the Savitzky-Golay smoother we used a cubic spline (`method = "spline"`) in the CPP function. In addition, outliers were automatically removed in the baseline region (Figure 6A and B). The background range was defined to be between the 1^{st} and 190^{th} data point (corresponds to baseline region between 0 and 52 minutes).

```
# Apply the CPP functions to pro-process the raw data.1) Baseline data to zero,
# 2) Smooth data with a spline, 3) Remove outliers in background range between
# entry 1 and 190. Assign the results of the analysis to the object 'res'.
res <- lapply(c(2, 4), function(i) {
  y.s <- CPP(C81[, i] / 60, C81[, i + 1],
             trans = TRUE,
             method = "spline",
             bg.outliers = TRUE,
             bg.range = c(1, 190))
  lines(C81[, i] / 60, y.s[["y.norm"]], type = "b", pch = 20, col = i - 1)
  # Use the th.cyc function to calculate the cycle threshold time (Cq.t).
  # The threshold signal level r was set to 0.05.
  paste(round(th.cyc(C81[, i] / 60, y.s[["y.norm"]], r = 0.05)[1], 2), "min")
})

# Add the cycle threshold time from the object 'res' to the plot.

abline(h = 0.05, lty = 2)
text(10, 0.55, "Cq.t:")
legend(10, 0.5, paste(c("D1: ", "D2: "), res), pch = 19, col = c(1, 3),
       bty = "n")
```

The pre-processed data were subjected to the analysis of the Cq_t values. It is important to note that the trend correction and proper baseline was a requirement for a sound calculation. We calculated Cq_t values of 70.18 minutes and 93.18 minutes for the stock (D1) and 1:10 (D2) diluted samples, receptively.

Case study four – digital PCR

We have developed the **dpcR** package for analysis and presentation of digital PCR experiments. The **dpcR** package can be used to build custom-made analysers and provides structures to be openly extended by the scientific community. Simulations and predictions of binomial and Poisson distributions, commonly used theoretical models of dPCR, statistical data analysis methods, plotting facilities and report generation tools are part of the package (Pabinger et al., 2014). Here, we show briefly a case study for the **dpcR**. Simulations are part in many educational curricula and support teaching greatly. In this case study, we mimicked an in-silico experiment for a droplet digital PCR similar to Figure 1. The aim was to asses the concentration of the template sample. The number of positive partitions (k), total number of partitions (n) and the size of the partition are the only data required for the analysis. The estimate of the mean number of template molecules per partition ($\hat{\lambda}$) was calculated using the following equation (Huggett et al., 2013):

$$\hat{\lambda} = -\ln(1 - \frac{k}{n}). \tag{1}$$

The average droplet volume in our experiment was assumed to be 5 nL. We counted $n = 16800$ droplets in total from which $k = 4601$ droplets were positive. The binomial distribution of positive and negative partitions was used to determine $\hat{\lambda}$. R allows both easy estimation of a density of the

parameter and calculation of confidence intervals using the Wilson method (Brown et al., 2001). The obtained mean number of template molecules per partition multiplied by the volume of the partitions ($16800 \cdot 5\ nL$) constitutes the sample concentration.

```
# Load the dpcR package for the analysis of the digital PCR experiment.
require(dpcR)

# Analysis of a digital PCR experiment. The density estimation.
# In our in-silico experiment we counted in total 16800 droplets (n).
# Thereof, 4601 were positive (k).

(dens <- dpcr_density(k = 4601, n = 16800, average = TRUE, methods = "wilson"))

# Let us assume, that every droplet has roughly a volume of 5 nL.
# The total concentration (and its confidence intervals) in molecules/ml is:
dens[4:6] / 5 * 1e-6
```

Selected functionality was implemented as interactive **shiny** (Chang et al., 2015) GUI application to make the software accessible for users who are not fluent in R and for experts who wish to automatize routine tasks. Details and examples of the **shiny** web application framework for R can be found at http://shiny.rstudio.com/. We implemented flexible user interfaces, which run the analyses and graphical representations into interactive web applications either as service on a web severer or on a local machine without knowledge of HTML or ECMAScript (see the **dpcR** manual). The interface is designed in a cascade workflow approach (Data import → Analysis → Output → Export) with interactive users choice on input data, methods and parameters using typical GUI elements such as sliders, drop-downs and text fields. An example can be found at https://michbur.shinyapps.io/dpcr_density/. This approach enables the automatized outputs of R objects in combined plots, tables and summaries.

Discussion and Conclusion

This study gave a brief introduction to the analysis of qPCR, qIA, MCA and dPCR experiments with R. In addition to this, we briefly referenced to a vast collection of additional packages available from CRAN and Bioconductor. The packages may be considered as the building blocks (libraries) to create what users want and need. We showed that automatized research with R offers powerful means for statistical analysis and visualization. The software is not tied to a vendor or specific application (e.g., chamber or droplet based digital PCR, capillary or plate qPCR). It should be quite easy even for an inexperienced user to define a workflow and to set up an environment for specific needs in a broad range of technical settings (Figure 8). R enforces no monolithic integration. We claim that the modular structure of R packages allows the user to perform flexible data analysis, adjusted to their needs and to design frameworks for high-throughput analysis. Furthermore, R enables the user to access and reuse code for the creation of reports in various formats (e.g., HTML, PDF). Most of the software is cross-platform open source software. Despite the fact that R is free of charge, it is quite possible to build commercial applications. The packages cover implementation of novel approaches and peer-reviewed analysis methods. R packages are an open environment to adopt to the growing knowledge in life sciences and medical sciences. Therefore, we argue that R may provide a structure for standardized nomenclature and severe as reference in qPCR and dPCR analysis. Speaking about openness, it needs to be emphasized that the main advantage of this software is its transparency at any time for anybody. Thus, it is possible to track numerical errors. A serve disadvantage of R is the lack of comprehensive GUIs for qPCR analysis. Others and we believe that a GUI is a key technology to spread the use of R in bioanalytical sciences. Currently, we are establishing the "pcRuniveRsum" (http://michbur.github.io/pcRuniveRsum/) as an on-line resource for the interested users. The command-line structure makes R "inaccessible" for many novices. We try to solve this problem with easily accessible GUIs (Rödiger et al., 2012). However, the work on this additions has been recently started and is still in progress.

Acknowledgments

Part of this work was funded by the BMBF InnoProfile-Transfer-Projekt 03 IPT 611X. We would like to thank the R community. Part of this work was funded by the Russian Ministry of Education and Science (project No. RFMEFI62114X0003) and with usage of scientific equipment of Center for collective use "Biotechnology" at All-Russia Research Institute of Agricultural Biotechnology. We

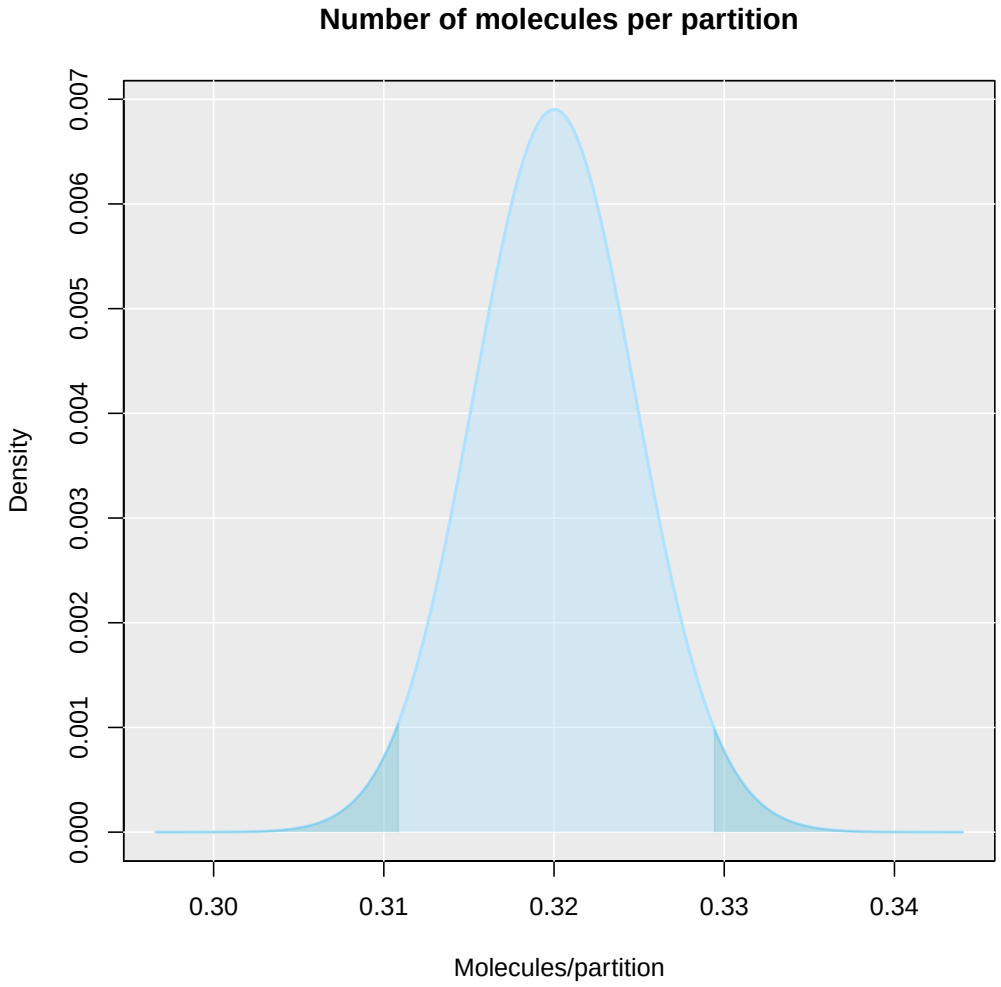

Figure 7: `dpcr_density` function from the **dpcR** package used for analysis of droplet digital PCR experiment. From 16800 counted droplets (n) 4601 were positive (k). The chart presents the distribution of mean number of template molecules per partition ($\hat{\lambda}$).

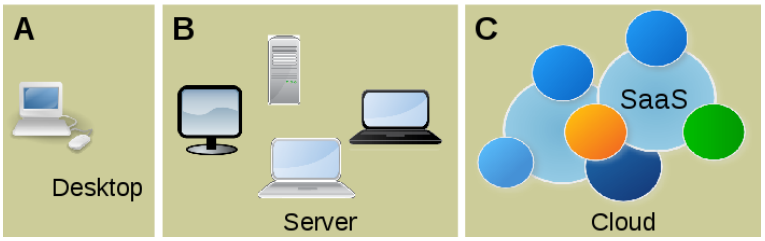

Figure 8: Deployment of R applications for the qPCR and dPCR experiments. **(A)** R is typically run from a desktop computer and operated by an GUI/IDE application such as Rstudio or RKWard. This approach provides a flexible workflow for individuals. **(B)** Another approach is to run R with specific applications on a local server. Such scenarios are useful for the deployment within research departments or cooperate units (Nolan and Temple, 2014). **(C)** Cloud computing (CC) provides shared and scalable computing capacity (e.g., computing capacity, application software) and storage capacity (e.g., databases) as a service to an individual user or a community Service categories include: Infrastructure-as-a-Service (IaaS), Platform-as-a-Service (PaaS) and Software-as-a-Service (SaaS) over a network. Providers of CC manage the infrastructure and resources to achieve coherence and economies of scale similar to a utility over a network (i.p., Internet; Ohri 2014).

would like to thank Mario Menschikowski (Technical University Dresden) for the droplet digital PCR samples.

Bibliography

M. G. Almiron, B. Lopes, A. L. C. Oliveira, A. C. Medeiros, and A. C. Frery. On the numerical accuracy of spreadsheets. *Journal of Statistical Software*, 34(4):1–29, 4 2010. URL http://www.jstatsoft.org/v34/i04. [p2]

R. Bååth. The state of naming conventions in R. *The R Journal*, 4(2):74–75, Dec. 2012. URL http://journal.r-project.org/archive/2012-2/RJournal_2012-2_Baaaath.pdf. [p3]

P. Biecek and M. Kosinski. *archivist: Tools for Storing, Restoring and Searching for R Objects*, 2015. URL http://CRAN.R-project.org/package=archivist. R package version 1.3. [p4]

K. A. Blagodatskikh, S. Roediger, and M. Burdukiewicz. *RDML: Importing Real-Time Thermo Cycler (qPCR) Data from RDML Format Files*, 2015. URL http://CRAN.R-project.org/package=RDML. R package version 0.8-4. [p2]

L. D. Brown, T. T. Cai, and A. Dasgupta. Interval estimation for a binomial proportion. *Statistical Science*, 16:101–133, 2001. [p13]

M. Burdukiewicz, S. Roediger, and B. Jacobs. *dpcR: Digital PCR Analysis*, 2015. URL http://CRAN.R-project.org/package=dpcR. R package version 0.1.4.0. [p2]

P. Burns. Spreadsheet addiction. online, 2014. URL http://web.archive.org/web/20141009042532/http://www.burns-stat.com/documents/tutorials/spreadsheet-addiction/. [p2]

S. A. Bustin, V. Benes, J. A. Garson, J. Hellemans, J. Huggett, M. Kubista, R. Mueller, T. Nolan, M. W. Pfaffl, G. L. Shipley, J. Vandesompele, and C. T. Wittwer. The MIQE guidelines: Minimum information for publication of quantitative real-time PCR experiments. 55(4):611–622. [p1]

W. Chang, J. Cheng, J. J. Allaire, Y. Xie, and J. McPherson. *shiny: Web Application Framework for R*, 2015. URL http://CRAN.R-project.org/package=shiny. R package version 0.12.1. [p13]

A. S. Devonshire, R. Sanders, T. M. Wilkes, M. S. Taylor, C. A. Foy, and J. F. Huggett. Application of next generation qPCR and sequencing platforms to mRNA biomarker analysis. *Methods (San Diego, Calif.)*, 59(1):89–100, Jan. 2013. [p1]

A. J. Durán, M. Pérez, and J. L. Varona. The misfortunes of a trio of mathematicians using computer algebra systems. Can we trust in them? *Notices of the American Mathematical Society*, 2014. [p2]

H. Dvinge and P. Bertone. HTqPCR: High-throughput analysis and visualization of quantitative real-time PCR data in R. *Bioinformatics*, 25(24):3325–3326, Dec. 2009. [p2]

C. Ekstrom, I. M. Skovgaard, and T. Martinussen. *kulife: Datasets and Functions from the (Now Non-Existing) Faculty of Life Sciences, University of Copenhagen*, 2013. URL http://CRAN.R-project.org/package=kulife. R package version 0.1-14. [p2]

F. C. Fang, R. G. Steen, and A. Casadevall. Misconduct accounts for the majority of retracted scientific publications. *Proceedings of the National Academy of Sciences of the United States of America*, 109(42):17028–17033, Oct. 2012. [p1]

C. Gandrud. *Reproducible Research with R and RStudio*. Chapman and Hall/CRC, July 2013. [p3]

R. C. Gentleman, V. J. Carey, D. M. Bates, B. Bolstad, M. Dettling, S. Dudoit, B. Ellis, L. Gautier, Y. Ge, J. Gentry, K. Hornik, T. Hothorn, W. Huber, S. Iacus, R. Irizarry, F. Leisch, C. Li, M. Maechler, A. J. Rossini, G. Sawitzki, C. Smith, G. Smyth, L. Tierney, J. Y. H. Yang, and J. Zhang. Bioconductor: Open software development for computational biology and bioinformatics. *Genome Biology*, 5(10):R80, 2004. [p2]

M. Gesmann and D. de Castillo. Using the Google visualisation API with R. *The R Journal*, 3(2):40–44, Dec. 2011. [p3]

M. Guescini, D. Sisti, M. B. L. Rocchi, L. Stocchi, and V. Stocchi. A new real-time PCR method to overcome significant quantitative inaccuracy due to slight amplification inhibition. *BMC Bioinformatics*, 9(326), July 2008. [p4]

H. Hofmann, A. Unwin, and D. Cook. Let graphics tell the story – Datasets in R. *The R Journal*, 5(1): 117–130, June 2013. [p3]

J. Huggett, J. O'Grady, and S. Bustin. How to make mathematics biology's next and better microscope. *Biomolecular Detection and Quantification*, 1(1):A1–A3, 2014. [p1]

J. F. Huggett, C. A. Foy, V. Benes, K. Emslie, J. A. Garson, R. Haynes, J. Hellemans, M. Kubista, R. D. Mueller, T. Nolan, M. W. Pfaffl, G. L. Shipley, J. Vandesompele, C. T. Wittwer, and S. A. Bustin. The digital MIQE guidelines: Minimum information for publication of quantitative digital PCR experiments. *Clinical Chemistry*, 59(6):892–902, June 2013. [p1, 12]

D. A. Khodakov and A. V. Ellis. Recent developments in nucleic acid identification using solid-phase enzymatic assays. *Microchimica Acta*, 181(13–14):1633–1646, 2014. [p1]

M. Kohl. *SLqPCR: Functions for Analysis of Real-Time Quantitative PCR Data at SIRS-Lab GmbH*. SIRS-Lab GmbH, Jena, 2007. URL www.sirs-lab.com. [p2]

T. J. Leeper. Archiving reproducible research and dataverse with R. *The R Journal*, 6(1):151–158, 2014. [p3]

S. Lefever, J. Hellemans, F. Pattyn, D. R. Przybylski, C. Taylor, R. Geurts, A. Untergasser, J. Vandesompele, and RDML Consortium. RDML: Structured language and reporting guidelines for real-time quantitative PCR data. *Nucleic Acids Research*, 37(7):2065–2069, Apr. 2009. [p1]

Z. Liu and S. Pounds. An R package that automatically collects and archives details for reproducible computing. *BMC Bioinformatics*, 15(1):138, May 2014. [p3]

T. Luo, L. Jiang, W. Sun, G. Fu, J. Mei, and Q. Gao. Multiplex real-time PCR melting curve assay to detect drug-resistant mutations of mycobacterium tuberculosis. *Journal of Clinical Microbiology*, 49 (9):3132–3138, Sept. 2011. [p7]

J. Mar. *qpcrNorm: Data-Driven Normalization Strategies for High-Throughput qPCR Data*, 2009. URL http://bioconductor.org. R package version 1.26.0. [p2]

M. V. Matz. *MCMC.qpcr: Bayesian Analysis of qRT-PCR Data*, 2015. URL http://CRAN.R-project.org/package=MCMC.qpcr. R package version 1.2. [p2]

M. N. McCall1, H. R. McMurray, H. Land, and A. Almudevar. On non-detects in qPCR data. *Bioinformatics*, 30(16):2310–2316, Aug. 2014. [p2]

B. D. McCullough and D. A. Heiser. On the accuracy of statistical procedures in Microsoft Excel 2007. *Computational Statistics & Data Analysis*, 52(10):4570–4578, June 2008. [p2]

P. Michna and M. Woods. RNetCDF – A package for reading and writing NetCDF datasets. *The R Journal*, 5(2):29–37, Dec. 2013. [p3]

C. A. Milbury, Q. Zhong, J. Lin, M. Williams, J. Olson, D. R. Link, and B. Hutchison. Determining lower limits of detection of digital PCR assays for cancer-related gene mutations. *Biomolecular Detection and Quantification*, 1(1):8–22, Sept. 2014. [p1]

A. A. Morley. Digital PCR: A brief history. *Biomolecular Detection and Quantification*, 1(1):1–2, Sept. 2014. [p1]

P. Murrell. It's not what you draw, it's what you don't draw. *The R Journal*, 4(2):13–18, Dec. 2012. [p3]

J. D. Neve and J. Meys. *unifiedWMWqPCR: Unified Wilcoxon-Mann-Whitney Test for qPCR Data*, 2014. URL http://bioconductor.org. R package version 1.4.0. [p2]

J. D. Neve, J. Meys, J.-P. Ottoy, L. Clement, and O. Thas. unifiedWMWqPCR: The unified Wilcoxon-Mann-Whitney test for analyzing RT-qPCR data in R. *Bioinformatics*, 30(17):2494–2495, 2014. [p2]

D. Nolan and D. L. Temple. *XML and Web Technologies for Data Sciences with R*. Springer-Verlag, New York, 2014. [p14]

J. Oh. Automatic conversion of tables to LongForm dataframes. *The R Journal*, 6(2):16–26, 2014. [p3]

A. Ohri. *R for Cloud Computing – An Approach for Data Scientists*. Springer-Verlag, New York, 2014. [p14]

S. Pabinger, S. Rödiger, A. Kriegner, K. Vierlinger, and A. Weinhäusel. A survey of tools for the analysis of quantitative PCR (qPCR) data. *Biomolecular Detection and Quantification*, 1, 2014. [p2, 12]

Y. Pan, X. Yan, and J. Li. *qPCR.CT: qPCR Data Analysis and Plot Package*, 2012. URL http://CRAN.R-project.org/package=qPCR.CT. R package version 1.1. [p2]

S. L. Pape. *EasyqpcR: EasyqpcR for Easy Analysis of Real-Time PCR Data at IRTOMIT-INSERM U1082*. IRTOMIT-INSERM U1082, 2012. URL http://irtomit.labo.univ-poitiers.fr/. [p2]

S. N. Peirson, J. N. Butler, and R. G. Foster. Experimental validation of novel and conventional approaches to quantitative real-time PCR data analysis. *Nucleic Acids Research*, 31(14):e73, July 2003. [p1]

J. R. Perkins, J. M. Dawes, C. Orengo, S. B. McMahon, D. L. Bennett, and M. Kohl. ReadqPCR and NormqPCR: R packages for the reading, quality checking and normalisation of RT-qPCR quantification cycle (Cq) data. *BMC Genomics*, 13(296), 2012. [p2]

R Core Team. *R Data Import/Export*. R Foundation for Statistical Computing, Vienna, Austria, 2015. URL http://www.R-project.org/. [p3]

C. Ritz and A.-N. Spiess. qpcR: an R package for sigmoidal model selection in quantitative real-time polymerase chain reaction analysis. *Bioinformatics*, 24(13):1549–1551, July 2008. PMID: 18482995. [p3]

S. Rödiger, T. Friedrichsmeier, P. Kapat, and M. Michalke. RKWard: A comprehensive graphical user interface and integrated development environment for statistical analysis with R. *Journal of Statistical Software*, 49(9):1–34, 2012. URL http://www.jstatsoft.org/v49/i09. [p3, 13]

S. Rödiger, A. Böhm, and I. Schimke. Surface melting curve analysis with R. *The R Journal*, 5(2):37–53, Dec. 2013a. [p2, 4, 7, 11]

S. Rödiger, P. Schierack, A. Böhm, J. Nitschke, I. Berger, U. Frömmel, C. Schmidt, M. Ruhland, I. Schimke, D. Roggenbuck, W. Lehmann, and C. Schröder. A highly versatile microscope imaging technology platform for the multiplex real-time detection of biomolecules and autoimmune antibodies. *Advances in Biochemical Engineering/Biotechnology*, 133:35–74, 2013b. [p1, 2, 10, 11]

S. Rödiger, C. Liebsch, C. Schmidt, W. Lehmann, U. Resch-Genger, U. Schedler, and P. Schierack. Nucleic acid detection based on the use of microbeads: A review. *Microchimica Acta*, 181(11–12): 1151–1168, 2014. [p1, 10]

S. Roediger. *MBmca: Nucleic Acid Melting Curve Analysis on Microbead Surfaces with R*, 2015. URL http://CRAN.R-project.org/package=MBmca. R package version 0.0.3-5. [p2]

S. Roediger and M. Burdukiewicz. *chipPCR: Toolkit of Helper Functions to Pre-Process Amplification Data*, 2014. URL http://CRAN.R-project.org/package=chipPCR. R package version 0.0.8-4. [p2, 4]

J. M. Ruijter, M. W. Pfaffl, S. Zhao, A. N. Spiess, G. Boggy, J. Blom, R. G. Rutledge, D. Sisti, A. Lievens, K. De Preter, S. Derveaux, J. Hellemans, and J. Vandesompele. Evaluation of qPCR curve analysis methods for reliable biomarker discovery: bias, resolution, precision, and implications. *Methods*, 59 (1):32–46, Jan. 2013. [p1, 2, 4]

J. M. Ruijter, P. Lorenz, J. M. Tuomi, M. Hecker, and M. J. B. v. d. Hoff. Fluorescent-increase kinetics of different fluorescent reporters used for qPCR depend on monitoring chemistry, targeted sequence, type of DNA input and PCR efficiency. *Microchimica Acta*, 181(13–14):1689–1696, Oct. 2014. [p1]

A.-N. Spiess. *qpcR: Modelling and Analysis of Real-Time PCR Data*, 2014. URL http://CRAN.R-project.org/package=qpcR. R package version 1.4-0. [p2]

A.-N. Spiess, C. Feig, and C. Ritz. Highly accurate sigmoidal fitting of real-time PCR data by introducing a parameter for asymmetry. *BMC Bioinformatics*, 9(1):221, Apr. 2008. [p3]

A.-N. Spiess, C. Deutschmann, M. Burdukiewicz, R. Himmelreich, K. Klat, P. Schierack, and S. Rödiger. Impact of smoothing on parameter estimation in quantitative DNA amplification. *Clinical Chemistry*, 61(2):379–388, 2015. [p2]

D. A. Swan. *DivMelt: HRM Diversity Assay Analysis Tool*, 2013. URL http://CRAN.R-project.org/package=DivMelt. R package version 1.0.3; with contributions from Craig A Magaret and Matthew M Cousins. [p2]

P. M. Valero-Mora and R. Ledesma. Graphical user interfaces for R. *Journal of Statistical Software*, 49(1): 1–8, June 2012. URL http://www.jstatsoft.org/v49/i01. [p3]

M. van der Loo. *settings: Software Option Settings Manager for R*, 2015. URL http://CRAN.R-project.org/package=settings. R package version 0.2.2. [p4]

E. Viturro, C. Altenhofer, B. Zölch, A. Burgmaier, I. Riedmaier, and M. W. Pfaffl. Microfluidic high-throughput reverse-transcription quantitative PCR analysis of liver gene expression in lactating animals. *Microchimica Acta*, 181(13-14):1725–1732, Oct. 2014. [p1]

J. Wu, R. Kodzius, W. Cao, and W. Wen. Extraction, amplification and detection of DNA in microfluidic chip-based assays. *Microchimica Acta*, 181(13-14):1611–1631, Oct. 2014. [p1]

M. Zeller, W.-C. L. A. Guazzelli, and G. Williams. PMML: An open standard for sharing models. *The R Journal*, 1(1):60–65, June 2009. [p3]

J. D. Zhang, R. Biczok, and M. Ruschhaupt. *ddCt: The ddCt Algorithm for the Analysis of Quantitative Real-Time PCR (qRT-PCR)*, 2015. URL http://bioconductor.org. R package version 1.22.0. [p2]

Stefan Rödiger (corresponding author)
Faculty of Natural Sciences
Brandenburg University of Technology Cottbus–Senftenberg
Senftenberg
Germany
Stefan.Roediger@hs-lausitz.de

Michał Burdukiewicz
University of Wroclaw
Faculty of Biotechnoloy
Department of Genomics
Wroclaw
Poland
michalburdukiewicz@gmail.com

Konstantin Blagodatskikh
All-Russia Research Institute of Agricultural Biotechnology
Center for collective use "Biotechnology"
Moscow
Russia
k.blag@yandex.ru

Peter Schierack
Faculty of Natural Sciences
Brandenburg University of Technology Cottbus–Senftenberg
Senftenberg
Germany
Peter.Schierack@hs-lausitz.de