# Short version of the vignette *End-to-end analysis of cell-based screens: from raw intensity readings to the annotated hit list*

Michael Boutros, Lígia Brás and Wolfgang Huber

April 7, 2016

## Contents

## 1 Introduction

This is a short version of the technical report *End-to-end analysis of cell-based screens: from raw intensity readings to the annotated hit list*, focusing on the essential steps necessary to run an analysis of a cell-based high-throughput screen (HTS), from raw intensity readings to an annotated hit list.

This report has been produced as a reproducible document [**?**]. It contains the actual computer instructions for the method it describes, and these in turn produce all results, including the figures and tables that are shown here. The computer instructions are given in the language R, thus, in order to reproduce the computations shown here, you will need an installation of R (version 2.3 or greater) together with a recent version of the package *cellHTS* and of some other add-on packages.

To reproduce the computations shown here, you do not need to type them or copy-paste them from the PDF file; rather, you can take the file *cellhts.Rnw* in the *doc* directory of the package, open it in a text editor, run it using the R command *Sweave*, and modify it to your needs.

For a more complete analysis, please refer to the complete vignette, which should be manually produced from the file *cellhtsComplete.Rnw* that can be found in the *scripts* directory of the package. The complete vignette accompanies the paper *Analysis of cell-based RNAi screens* by Michael Boutros, Lígia Brás and Wolfgang Huber [**?**], and includes sections exemplifying how to add more annotation from public databases, how to perform an analysis

| Filename | Plate | Replicate |
|---|---|---|
| FT01-G01.txt | 1 | 1 |
| FT01-G02.txt | 1 | 2 |
| FT02-G01.txt | 2 | 1 |
| FT02-G02.txt | 2 | 2 |
| FT03-G01.txt | 3 | 1 |
| ... | ... | ... |

Table 1: Selected lines from the example plate list file `Platelist.txt`.

for Gene Ontology categories, and compares the obtained results with previously reported ones.

First, we load the package.

```
> library("cellHTS")
```

## 2 Reading the intensity data

We consider a cell-based screen that was conducted in microtiter plate format, where a library of double-stranded RNAs was used to target the corresponding genes in cultured *Drosophila* $Kc_{167}$ cells [**?**]. Each of the wells in the plates contains either a gene-specific probe, a control, or it can be empty. The experiments were done in duplicate, and the viability of the cells after treatment was recorded by a plate reader measuring luciferase activity, which is indicative of ATP levels. Although this set of example data corresponds to a single-channel screening assay, the *cellHTS* package can also deal with cases where there are readings from more channels, corresponding to different reporters. Usually, the measurements from each replicate and each channel come in individual result files. The set of available result files and the information about them (which plate, which replicate, which channel) is contained in a spreadsheet, which we call the *plate list file*. This file should contain the following columns: *Filename*,*Plate*, and *Replicate*. The last two columns should be numeric, with values ranging from 1 to the maximum number of plates or replicates, respectively. The first few lines of an example plate list file are shown in Table **??**.

The first step of the analysis is to read the plate list file, to read all the intensity files, and to assemble the data into a single R object that is suitable for subsequent analyses. The main component of that object is one

big table with the intensity readings of all plates, channels, and replicates. We demonstrate the R instructions for this step. First we define the path where the input files can be found.

```
> experimentName = "KcViab"
> dataPath=system.file(experimentName, package="cellHTS")
```

In this example, the input files are in the `KcViab` directory of the *cellHTS* package. To read your own data, modify `dataPath` to point to the directory where they reside. We show the names of 12 files from our example directory:

```
> dataPath

[1] "/private/tmp/RtmpCLB94g/Rinst1027debea9c/cellHTS/KcViab"

> rev(dir(dataPath))[1:12]

 [1] "Screenlog.txt"          "Platelist.txt"          "Plateconf.txt"
 [4] "GeneIDs_Dm_HFA_1.1.txt" "FT57-G02.txt"           "FT57-G01.txt"
 [7] "FT56-G02.txt"           "FT56-G01.txt"           "FT55-G02.txt"
[10] "FT55-G01.txt"           "FT54-G02.txt"           "FT54-G01.txt"
```

and read the data into the object `x`

```
> x = readPlateData("Platelist.txt", name=experimentName, path=dataPath)

> x

cellHTS object of name 'KcViab'
57 plates with 384 wells, 2 replicates, 1 channel. State:
configured normalized     scored  annotated
     FALSE      FALSE      FALSE      FALSE
```

The plate format used in the screen (96-well or 384-well plate design) is automatically determined from the raw intensity files, when calling the *readPlateData* function.

## 3 The *cellHTS* class and reports

The basic data structure of the package is the class *cellHTS*. In the previous section, we have created the object `x`, which is an instance of this class. All subsequent analyses, such as normalization, gene selection and annotation,

| Batch | Well | Content |
|---|---|---|
| 1 | B01 | neg |
| 1 | B02 | pos |
| 1 | B03 | sample |
| 1 | B04 | sample |
| ... | ... | ... |

Table 2: Selected lines from the example plate configuration file `Plateconf.txt`.

will add their results into this object. Thus, the complete analysis project is contained in this object, and a complete dataset can be shared with others and stored for subsequent computational analyses in the form of such an object. In addition, the package offers export functions for generating human-readable reports, which consist of linked HTML pages with tables and plots. The final scored hit list is written as a tab-delimited format suitable for reading by spreadsheet programs.

To create a report, use the function *writeReport*. It will create a directory of the name given by `x$name` in the working directory. Alternatively, the argument `outdir` can be specified to direct the output to another directory [1]

```
> out = writeReport(x)
```

It can take a while to run this function, since it writes a large number of graphics files. After this function has finished, the index page of the report will be in the file indicated by the variable `out`,

```
> out
```

and you can view it by directing a web browser to that file.

```
> browseURL(out)
```

# 4 Annotating the plate results

The next step of the analysis is to annotate the measured data with information on controls and to flag invalid measurements. The software expects

[1] To reduce the amount of time this vignette takes to be created, the given code for generating the quality reports will not be evaluated when calling *Sweave*. You can easily change this by editing the *cellhts.Rnw* file.

| Filename | Well | Flag | Comment |
|---|---|---|---|
| FT06-G01.txt | A01 | NA | Contamination |
| FT06-G02.txt | A01 | NA | Contamination |
| FT06-G01.txt | A02 | NA | Contamination |
| ... | ... | ... | ... |

Table 3: Selected lines from the example screen log file `Screenlog.txt`.

the information on the controls in a so-called *plate configuration file* (see Section **??**). This is a tab-delimited file with one row per well. Selected lines of this file are shown in Table **??**.

Individual measurements can be flagged as invalid in the so-called *screen log file* (see Section **??**). The first 5 lines of this file are shown in Table **??**.

The *screen description* file contains a general description of the screen, its goal, the conditions under which it was performed, references, and any other information that is pertinent to the biological interpretation of the experiments.

We now apply this information to the data object `x`.

```
> x = configure(x, "Plateconf.txt", "Screenlog.txt",
+         "Description.txt", path=dataPath)
```

Note that the function *configure*[2] takes `x`, the result from Section **??**, as an argument, and we then overwrite `x` with the result of this function. If no screen log file is available for the experiment, the argument `logFile` of the function *configure* should be omitted.

## 4.1 Format of the plate configuration file

The software expects this to be a rectangular table in a tabulator delimited text file, with mandatory columns *Batch*, *Well*, *Content*. The *Batch* column allows to have different plate configurations (see Section **??**). The *Well* column contains the name of each well of the plate, in letter-number format (in this case, `A01` to `P24`). As the name suggests, the *Content* column provides the content of each well in the plate (here referred to as the *well annotation*). Mainly, this annotation falls into four categories: empty wells, wells containing genes of interest, control wells, and wells containing other things that do not fit in the previous categories. The first two types of wells should

[2]More precisely, *configure* is a method for the S3 class *cellHTS*.

be indicated in the *Content* column of the plate configuration file by *empty* and *sample*, respectively, while the last type of wells should be indicated by *other*. The designation for the control wells in the *Content* conlumn is more flexible. By default, the software expects them to be indicated by *pos* (for positive controls), or *neg* (for negative controls). However, other names are allowed, given that they are specified by the user whenever necessary (for example, when calling the *writeReport* function). This versatility for the control wells' annotation is justified by the fact that,sometimes, multiple positive and/or negative controls can be employed in a given screen, making it useful to give different names to the distinct controls in the *Content* column. Moreover, this versatility is also required in multi-channel screens for which we frequently have reporter-specific controls. Note that the well annotations mentioned above are used by the software in the normalization, quality control, and gene selection calculations. Data from wells that are annotated as *empty* are ignored, i. e. they are set to `NA`. Here we look at the frequency of each well annotation in the example data:

```
> table(x$plateConf$Content)

   neg  other    pos sample
     1      2      1    380
```

Another case is when different types of positive controls are used for the screening, that is *activator* and *inhibitor* compounds. The vignette *Analysis of two-way cell-based assays* accompanying this package explains how such screens can be handled using *cellHTS* package.

### 4.1.1 Multiple plate configurations

Although it is good practice to use the same plate configuration for the whole experiment, sometimes this does not work out, and there are different parts of the experiment with different plate configurations. It is possible to specify multiple plate configurations simply by appending them to each other in the plate configuration file, and marking them with different numbers in the column *Batch*.

Note that replicated experiments per plate have to use the same plate configuration.

## 4.2 Format of the screen log file

The screen log file is a tabulator delimited file with mandatory columns *Filename*, *Well*, *Flag*. In addition, it can contain arbitrary optional columns.

Each row corresponds to one flagged measurement, identified by the filename and the well identifier. The type of flag is specified in the column *Flag*. Most commonly, this will have the value "NA", indicating that the measurement should be discarded and regarded as missing.

# 5 Normalization and summarization of replicates

The function *normalizePlates* can be called to adjust for plate effects. Its parameter `normalizationMethod` allows to choose between different types of normalization. For example, if it is set to `"median"`, the function *normalizePlates* adjusts for plate effects by dividing each value in each plate by the median of values in the plate:

$$x'_{ki} = \frac{x_{ki}}{M_i} \qquad \forall k, i \tag{1}$$

$$M_i = \underset{m \in \text{ samples}}{\text{median}} \; x_{mi} \tag{2}$$

where $x_{ki}$ is the raw intensity for the $k$-th well in the $i$-th replicate file, and $x'_{ki}$ is the corresponding normalized intensity. The median is calculated across the wells annotated as *sample* in the $i$-th result file. This is achieved by calling

```
> x = normalizePlates(x, normalizationMethod="median")
```

after which the normalized intensities are stored in the slot `x$xnorm`. This is an array of the same size as `x$xraw`.

We can now summarize the replicates, calculating a single score for each gene. One option would be to take the root mean square of the values from the replicates:

$$z_{ki} = \pm \frac{x'_{ki} - \hat{\mu}}{\hat{\sigma}} \tag{3}$$

$$z_k = \sqrt{\frac{1}{n_{\text{rep}_k}} \sum_{r=1}^{n_{\text{rep}_k}} z_{kr}^2}. \tag{4}$$

Before summarizing the replicate, we standardize the values for each replicate experiment using Equation (**??**). Here $\hat{\mu}$ and $\hat{\sigma}$ are estimators of location and scale of the distribution of $x'_{ki}$ taken across all plates and wells of a given replicate experiment. We use robust estimators, namely, median and median absolute deviation (MAD). Moreover, we only consider the wells

containing "sample" for estimating $\hat{\mu}$ and $\hat{\sigma}$. As the values $x'_{ki}$ were obtained using plate median normalization (**??**), it holds that $\hat{\mu} = 1$. The symbol $\pm$ indicates that we allow for either plus or minus sign in equation (**??**); the minus sign can be useful in the application to an inhibitor assay, where an effect results in a decrease of the signal and we may want to see this represented by a large $z$-score. Then, in Equation (**??**), the summary is taken over all the $n_{\text{rep}_k}$ replicates of probe $k$.

Depending on the intended stringency of the analysis, other plausible choices of summary function between replicates are the minimum, the maximum, and the mean. In the first case, the analysis would be particularly conservative: all replicate values have to be high in order for $z_k$ to be high. For the cases where both sides of the distribution of $z$-score values are of interest, alternative summary options for the replicates are to select the value closest to zero (conservative approach) by setting `summary`='closestToZero' or the value furthest from zero (`summary`='furthestFromZero'). In order to compare our results with those obtained in the paper of Boutros *et al.* [**?**], we choose to consider the mean as a summary:

```
> x = summarizeReplicates(x, zscore="-", summary="mean")
```

The resulting single $z$-score value per probe will be stored in the slot `x$score`. Boxplots of the $z$-scores for the different types of probes are shown in Figure **??**.

```
> ylim = quantile(x$score, c(0.001, 0.999), na.rm=TRUE)
> boxplot(x$score ~ x$wellAnno, col="lightblue", outline=FALSE, ylim=ylim)
```

### 5.1 Alternative processing strategies

The HTML quality report will consider the values in the slot `x$xnorm` for the calculation of its quality metrics. In the example above, `x$xnorm` contains the data after plate median normalization, but before calculation of the $z$-scores and the multiplication by $-1$. The package *cellHTS* allows some flexibility with respect to these steps. We can already calculate the $z$-scores and multiply by $-1$ in the function *normalizePlates*, and then do the summarization between replicates, by calling the function *summarizeReplicates* without the argument `zscore`.

```
> xalt = normalizePlates(x, normalizationMethod="median", zscore="-")
> xalt = summarizeReplicates(xalt, summary="mean")
```

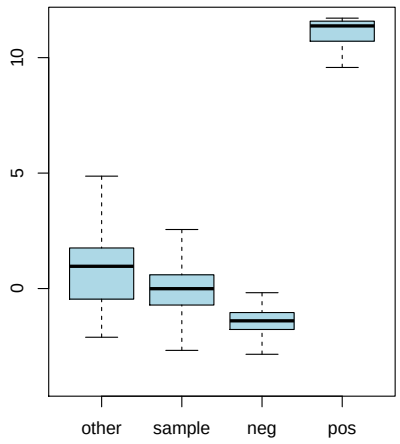


Figure 1: Boxplots of $z$-scores for the different types of probes.

It is easy to define alternative normalization methods, for example, to adjust for additional experimental biases besides the plate effect. You might want to start by taking the source code of *normalizePlates* as a template.

# 6 Annotation

Up to now, the assayed genes have been identified solely by the identifiers of the plate and the well that contains the probe for them. The *annotation file* contains additional annotation, such as the probe sequence, references to the probe sequence in public databases, the gene name, gene ontology annotation, and so forth. Mandatory columns of the annotation file are *Plate*, *Well*, and *GeneID*, and it has one row for each well. The content of the *GeneID* column will be species- or project-specific. The first 5 lines of the example file are shown in Table **??**, where we have associated each probe with CG-identifiers for the genes of *Drosophila melanogaster*.

```
> x = annotate(x, geneIDFile="GeneIDs_Dm_HFA_1.1.txt", path=dataPath)
```

An optional column named *GeneSymbol* can be included in the *annotation file*, and its content will be displayed by the tooltips added to the plate plots

| Plate | Well | HFAID | GeneID |
|---|---|---|---|
| 1 | A03 | HFA00274 | CG11371 |
| 1 | A04 | HFA00646 | CG31671 |
| 1 | A05 | HFA00307 | CG11376 |
| 1 | A06 | HFA00324 | CG11723 |
| ... | ... | ... | ... |

Table 4: Selected lines from the example gene ID file `GeneIDs_Dm_HFA_1.1.txt`.

and screen-wide plot, in the HTML quality report (see Section **??**).

# 7 Report

We have now completed the analysis tasks: the dataset has been read, configured, normalized, scored, and annotated:

```
> x

cellHTS object of name 'KcViab'
57 plates with 384 wells, 2 replicates, 1 channel. State:
configured normalized     scored  annotated
      TRUE       TRUE       TRUE       TRUE
```

We can now save the data set to a file.

```
> save(x, file=paste(experimentName, ".rda", sep=""), compress=TRUE)
```

The dataset can be loaded again for subsequent analysis, or passed on to others. To produce a comprehensive report, we can call the function *writeReport* again,

```
> out = writeReport(x, force=TRUE,
+   plotPlateArgs   = list(xrange=c(0.5, 1.5)),
+   imageScreenArgs = list(zrange=c( -2, 6.5), ar=1))
```

and use a web browser to view the resulting report

```
> browseURL(out)
```

The report contains a quality report for each plate, and also for the whole screening assays. The per-plate HTML reports display the scatterplot between duplicated plate measurements, the histogram of the normalized signal intensities for each replicate, and plate plots representing, in a false color scale, the normalized values of each replicate, and the standard deviation between replicate measurements at each plate position. It also reportes the Spearman rank correlation coefficient between duplicates, and the dynamic range, calculated as the ratio between the geometric means of the positive and negative controls. If different positive controls were specified at the configuration step and when calling *writeReport*, the dynamic range is calculated separately for the distinct positive controls, since different positive controls might have different potencies.

The experiment-wide HTML report presents, for each replicate, the the boxplots with raw and normalized intensities for the different plates, and two plots for the controls: one showing the signal from positive and negative controls at each plate, and another plot displaying the distribution of the signal from positive and negative controls, obtained from kernel density estimates. The latter plot further gives the $Z'$-factor determined for each experiment (replicate) using the negative controls and each different type of positive controls [**?**], as a measure to quantify the distance between their distributions. The experiment-wide report also shows a screen-wide plot with the $z$-scores in every well position of each plate. This plot, as well as the plate plots of the per-plate reports contain tooltips (information popup boxes) dispaying the annotation information at each position within the plates. If the `cellHTS` object has not been annotated yet, the annotation information shown by the tooltips is simply the well identifiers. For an annotated `cellHTS` object, if an optional column called *GeneSymbol* was included in the *annotation file* (see Section **??**), and therefore is present in `x$geneAnno`), its content is used for the tooltips. Otherwise, the content of `x$geneAnno$GeneID` is considered.

The screen-wide image plot can also be produced separately using the function *imageScreen* given in the *cellHTS* package. This might be useful if we want to select the best display for our data, namely, the aspect ratio for the plot and/or the range of $z$-score values to be mapped into the color scale. These can be passed to the function's arguments `ar` and `zrange`, respectively. For example,

```
> imageScreen(x, ar=1, zrange=c(-3,4))
```

It should be noted that the per-plate and per-experiment quality reports are constructed based on the content of `x$xnorm`, if it is present in the `x`

object. Otherwise, it uses the content given in the slot `x$xraw`. In the case of dual-channel experiments, the `x$xnorm` slot could also contain the ratio between the intensities in two different channels, etc. The main point that we want to highlight is that `x$xnorm` should contain the data that we want to visualize in the HTML quality reports. On the other hand, `x$score` should always contain the final list of scored probes (one value per probe).

The quality report produced by *writeReport* function has also a link to a file called *topTable.txt* that contains the list of scored probes ordered by decreasing $z$-score values. This file has one row for each well and plate, and for the present example data set, it has the following columns:

- `plate`;
- `position` gives the position of the well in the plate (runs from 1 to the total number of wells in the plate);
- `score` corresponds to the score calculated for the probe (content of `x$score`);
- `wellAnno` corresponds to the well annotation (as given by the plate configuration file;
- `normalized_r1_ch1` and `normalized_r2_ch1` give the normalized intensities for replicate 1 and replicate 2, respectively ('ch' refers to channel). This corresponds to the content of `x$xnorm`;
- `xrawAnno_r1_ch1` and `xrawAnno_r2_ch1` give the final well annotation for replicate 1 and 2, respectively. It combines the information given in the plate configuration file with the values in `x$xraw`, in order to have into account the wells that have been flagged either by the screen log file, or manually by the user during the analysis. These flagged wells appear with the annotation *flagged*.
- `raw_r1_ch1` and `raw_r2_ch1` contain the raw intensities for replicate 1 and replicate 2, respectively (content of `x$xraw`);
- `median_ch1` corresponds to the median of raw measurements across replicates;
- `diff_ch1` gives the difference between replicated raw measurements (only given if the number of replicates is equal to two);
- `average_ch1` corresponds to the average between replicated raw intensities (only given if the number of replicates is higher than two);

- `raw/PlateMedian_r1_ch1` and `raw/PlateMedian_r2_ch1` give the ratio between each raw measurement and the median intensity in each plate for replicate 1 and replicate 2, respectively. The plate median is determined for the raw intensities, using exclusively the wells annotated as "sample".

Additionally, if `x` has been annotated (as in the present case), it also contains the data given in the original gene anotation file that was stored in `x$geneAnno`.

## 7.1 Exporting data to a tab-delimited file

The *cellHTS* package contains a function called *writeTab* to save `x$xraw` and, if available, `x$xnorm` data from a *cellHTS* object to a tab-delimited file to a file. The rows of the file are sorted by plate and well, and there is one row for each plate and well. Its columns correspond to the content of `x$geneAnno` (that is, the gene annotation information), together with the raw measurements, and if available, the normalized intensities for each replicate and channel. The name for the columns containing the raw intensities starts with "R" and is followed by the replicate identifier "r", and by the channel identifier "c". For example, `Rr2c1` refers to the raw data for replicate 2 in channel 1. For the normalized data, the column names start with "N" instead of "R".

```
> writeTab(x, file="Data.txt")
```

Since you might be interestered in saving other values to a tab delimited file, below we demonstrate how you can create a matrix with the ratio between each raw measurement and the plate median, together with the gene and well annotation, and export it to a tab-delimited file using the function *write.tabdel* [3] also provided in the *cellHTS* package.

```
> # determine the ratio between each well and the plate median
> y = array(as.numeric(NA), dim=dim(x$xraw))
> nrWell = dim(x$xraw)[1]
> for(p in 1:(dim(x$xraw)[2])) {
+   samples = (x$wellAnno[(1:nrWell)+nrWell*(p-1)]=="sample")
+   y[, p, , ] = apply(x$xraw[, p, , , drop=FALSE], 3:4,
+      function(w) w/median(w[samples], na.rm=TRUE)) }
```

[3] This function is a wrapper of the function *write.table*, whereby you just need to specify the name of the data object and the file

```
> y=signif(y, 4)
> out = matrix(y, nrow=prod(dim(y)[1:2]), ncol=dim(y)[3:4])
> out = cbind(x$geneAnno, x$wellAnno, out)
> colnames(out) = c(names(x$geneAnno), "wellAnno",
+ sprintf("Well/Median_r%d_ch%d", rep(1:dim(y)[3], dim(y)[4]),
+ rep(1:dim(y)[4], each=dim(y)[3])))
> write.tabdel(out, file="WellMedianRatio.txt")
```

At this point we are finished with the basic analysis of the screen. As one example for how one could continue to further mine the screen results for biologically relevant patterns, we demonstrate an application of category analysis in the complete vignette, which is given as a zipped PDF file in the *doc* directory of the package, or can otherwise be manually produced from the file *cellhtsComplete.Rnw* that resides in the *scripts* directory of the package.

# 8 Appendix: Data transformation

An obvious question is whether to do the statistical analyses on the original intensity scale or on a transformed scale such as the logarithmic one. Many statistical analysis methods, as well as visualizations work better if (to sufficient approximation)

- replicate values are normally distributed,
- the data are evenly distributed along their dynamic range,
- the variance is homogeneous along the dynamic range [**?**].

Figure **??** compares these properties for untransformed and log-transformed normalized data, showing that the difference is small. Intuitively, this can be explained by the fact that for small $x$,

$$\log(1 + x) \approx x$$

and that indeed the range of the untransformed data is mostly not far from 1. Hence, for the data examined here, the choice between original scale and logarithmic scale is one of taste, rather than necessity.

```
> library("vsn")
> par(mfcol=c(3,2))
> myPlots=function(z, main=NULL, ...) {
```

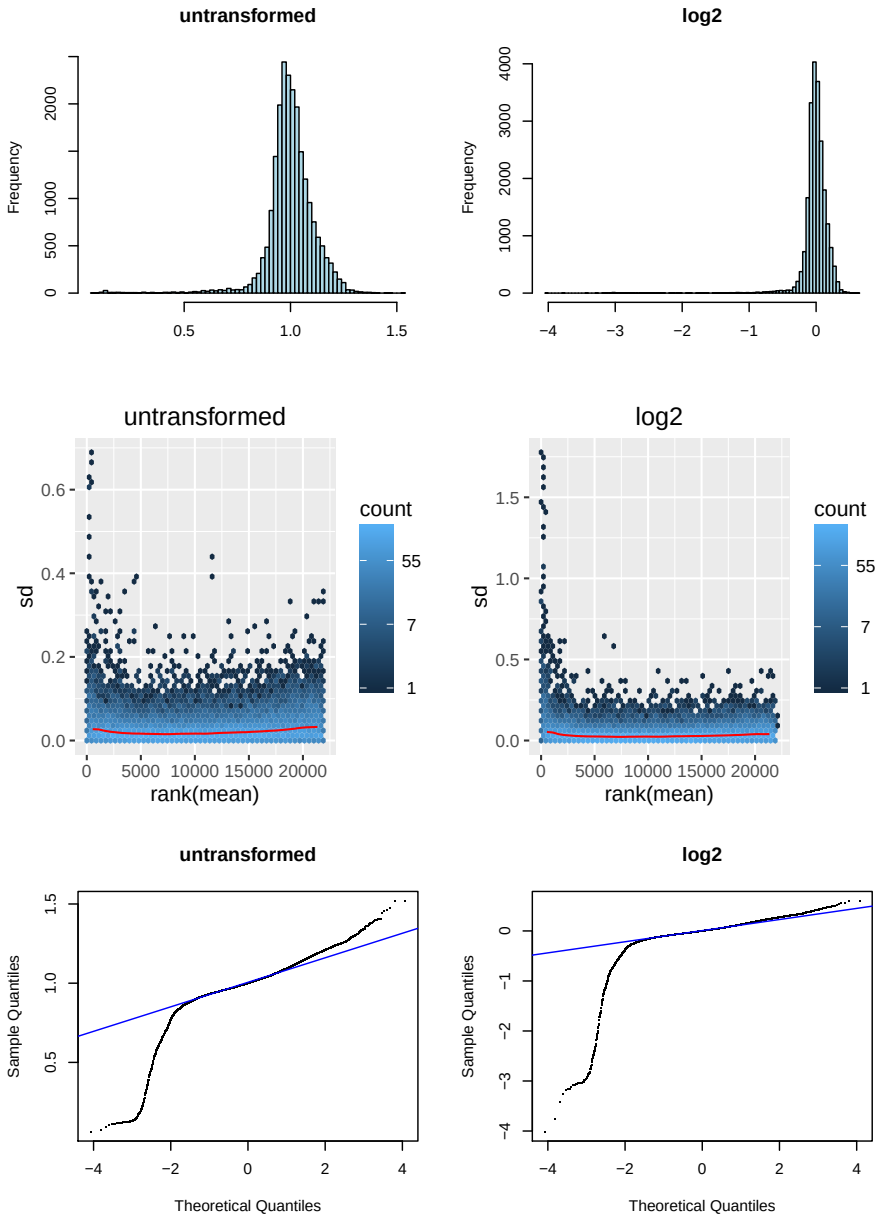


Figure 2: Comparison between untransformed (left) and logarithmically (base 2) transformed (right), normalized data. Upper: histogram of intensity values of replicate 1. Middle: scatterplots of standard deviation versus mean of the two replicates. Bottom: Normal quantile-quantile plots.

```
+   hist(z[,1], 100, col="lightblue", xlab="", main=main, ...)
+   plot(1L, type="n", xlab="", ylab="", asp=1, ...)
+   plot(meanSdPlot(z, plot=FALSE)$gg + ggplot2::ggtitle(main), vp=gridBase::baseView
+   qqnorm(z[,1], pch='.', main=main, ...)
+   qqline(z[,1], col='blue')
+ }
> dv = matrix(x$xnorm, nrow=prod(dim(x$xnorm)[1:2]), ncol=dim(x$xnorm)[3])
> myPlots(dv, main="untransformed")
> xlog = normalizePlates(x, normalizationMethod="median", transform=log2)
> dvlog = matrix(xlog$xnorm, nrow=prod(dim(xlog$xnorm)[1:2]), ncol=dim(xlog$xnorm)[3]
> myPlots(dvlog, main="log2")
```

# 9 Session info

This document was produced using:

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

- R version 3.2.4 (2016-03-10), `x86_64-apple-darwin13.4.0`
- Locale: `C/en_US.UTF-8/en_US.UTF-8/C/en_US.UTF-8/en_US.UTF-8`
- Base packages: base, datasets, grDevices, graphics, grid, methods, parallel, stats, utils
- Other packages: Biobase 2.30.0, BiocGenerics 0.16.1, cellHTS 1.40.2, hexbin 1.27.1, vsn 3.38.0
- Loaded via a namespace (and not attached): AnnotationDbi 1.32.3, BiocInstaller 1.20.1, DBI 0.3.1, DEoptimR 1.0-4, IRanges 2.4.8, MASS 7.3-45, RColorBrewer 1.1-2, RSQLite 1.0.0, Rcpp 0.12.4, S4Vectors 0.8.11, XML 3.98-1.4, affy 1.48.0, affyio 1.40.0, annotate 1.48.0, cluster 2.0.3, colorspace 1.2-6, digest 0.6.9, genefilter 1.52.1, ggplot2 2.1.0, gridBase 0.4-7, gtable 0.2.0, labeling 0.3, lattice 0.20-33, limma 3.26.9, munsell 0.4.3, mvtnorm 1.0-5, pcaPP 1.9-60, plyr 1.8.3, prada 1.46.0, preprocessCore 1.32.0, robustbase 0.92-5, rrcov 1.3-11, scales 0.4.0, splines 3.2.4, stats4 3.2.4, survival 2.38-3, tools 3.2.4, xtable 1.8-2, zlibbioc 1.16.0