OperaMate: data importing, processing and analysis for Opera High Content Screening System

Chenglin Liu

Shanghai Jiaotong University,
Shanghai, China
cliu@sjtu.edu.cn

Yixue Li

Shanghai Jiaotong Univeristy,
Shanghai, China
yxli@sibs.ac.cn

October 14, 2015

Contents

1 Introduction

The *OperaMate* is intended to analyze intensity data derived from the images from PerkinElmer's Opera High Content Screening System (http://www.perkinelmer.com/pages/020/cellularimaging/products/opera.xhtml) and intepreted by Columbus Image Data Storage and Analysis System (http://www.perkinelmer.com/pages/020/cellularimaging/products/columbus.xhtml). PerkinElmer's Opera High Content Screening System is the state-of-the-art confocal microplate imaging solution for high throughput screening. The system works with complex disease models and offers various solutions like protein expression, RNAi screening. Its compatible tool Columbus image analysis system exports intensity data by analyzing the Opera images, but lacks further processing and analysis to uncover the biological meaning underlying the

image data. Although some existing tool like cellHTS [1] can be used to process the intensity data, it requires massive configurations and data format modifications. Hence, we develop an R [2] package `OperaMate` especially to process the intensity data exported by Columbus system, which can fulfills the procedure of data importing, processing and analysis in an easy and efficient way.

2 Getting Started

OperaMate integrates the entire process of data importing, processing and analysis into one function, which is easy to operate for the users who are new to the R language. However, it is also very convinient to customize each step according to the pipeline built by this function, checking the immediate results of each step, and re-performing a specific step with different parameters to achieve the desire of the users.

2.1 Configration

OperaMate requires two reference tables and a registered email address at DAVID WebService [3] for data processing.

platemap data.frame, the experiment information of each Columbus analysis report. This table is required only if the report formats are not standarded. See Section 4.1 for more information.
Required column names of `platemap`:
- file.name: character, the name of the report.
- format.type: character, only "Tab" and "Matrix" are supported in this version.
- Barcode: character, the barcode of the plates.
- rep.id: character, the ID to distinguish the replicated plates.
- path: character, the full path of the report.

Refer to ?platemap for more information.

genemap data.frame, the gene names and information of each well. The correponding file is generated during the design of assays.
Required column names of `genemap`:
- Barcode: character, the barcode of the plates.
- Well: character, the well ID.
- GeneSymbol: character, the annotated gene names of the well.
- Gene.ID: character, Entrez gene ID.

Refer to ?genemap for more information.

email the email registered at DAVID WebService (http://david.abcc.ncifcrf.gov/content.jsp?file=WS.html). See Section 5 for more information.

2.2 Running OperaMate

```
> library(OperaMate)
> data(platemap)
> data(genemap)
> platemap$path <- file.path(system.file("Test",package = "OperaMate"),
+                            platemap$path)
> (outpath <- tempdir())
```

```
[1] "/tmp/RtmpBbCBuN"
```

```
> operaReport <- operaMate(cellformat = "Others",
+                          platemap = platemap,genemap = genemap,
+                          Prefix.Barcode = "DSIMGA",outpath=outpath,
+                          expwell = paste0(rep(LETTERS[2:15],each=20),
+                            rep(formatC(3:22,width=2,flag=0),times=14)),
```

```
+                           ctrwell = c("C23","E23","G23"),
+                           norm.method="Both",BatcheffectSample="exp",
+                           email = "cliu@sjtu.edu.cn",david.type="all",
+                           Sig.method="ktsd",width=960,height=960)
> names(operaReport)
```

```
[1] "lstCells"  "data.anno"
```

A brief description of the parameters are as follows:

1. `cellformat`, format of the to be processed reports. See Section 4.1 for more information.
2. `datapath` or `platemap`, character of the dictionary of the files or a platemap table. See Section 4.1 for more information.
3. `genemap`, genemap table.
4. `outpath`, the dictionary of the output reports and figures.
5. `Prefix.Barcode`, character of the prefix of the barcode in the genemap, e.g. DSIMGA.
6. `ctrwell`, character of the control well IDs.
7. `expwell`, character of the sample well IDs.
8. `email`, the email registered for the DAVID WebService.

See the mannual for more details about this function.

2.3 Description of output

All the reports and figures are located in the outpath.

1. OperaMateReport.txt: the analysis report.
2. DAVID-*.txt: the DAVID functional analysis reports.
3. *.before.after.normalization.png: the data comparison before and after normalization.
4. *.vocanoPlot.png: the volcano plot which highlights the detected hits.
5. Statistics.png: the histogram of the number of the hits.
6. *.sd2mean.png: the figure in the quality control step. See Section 4.3 for more details.
7. *.Distribution.png: the figure in the hit identification step. See Section 5 for more details.

3 Class declaration

3.1 The expData class

Each `expData` object stores data of one imaging analysis report of the Opera system generatd by Columbus^{TM} Image Data Storage and Analysis System. The report includes different types of data of one plate, distinguished by the paramters. The format of the report is set in the analysis system. *OperaMate* supports two most popular formats by now: the matrix and the table. The class requires the following information:

1. `name`, name of the plate
2. `path`, path of the importing file
3. `rep.id`, replicate ID
4. `exp.id`, barcode of the experiment
5. `format`, report format
   (a) `Matrix`: matrix format in the Columbus analysis system
   (b) `Tab`: table format in the Columbus analysis system

An example of creating a new expData object is as follows:

```
> (onePlate <- expData(name = "DSIMGA02-s1",
+                      path = file.path(
```

```
+                         system.file("Test",package = "OperaMate"),
+                         "Matrix", "130504-s1-02.txt" ),
+                       rep.id = "s1", exp.id = "DSIMGA02",
+                       format = "Matrix"))
```

```
An object of expData class.
Name: DSIMGA02-s1.
Path: /private/tmp/Rtmpdrp0n5/Rinst3e83e1ccb6a/OperaMate/Test/Matrix/130504-s1-02.txt.
Data:  list()
Wells:  chr(0)
```

However, it is highly recommended to import all files using the function `loadAll`. Details will be referred in Section 4.1.

3.2 The cellData class

An object of `expData` class stores all data corresponding to one parameter of the reports. It organizes the data as a matrix with rows are well IDs and columns the plate IDs. This class requires the following information:

1. `name`, one parameter of the report
2. `ctwell`, the well IDs of the controls, e.g. B05
3. `expwell`, the well IDs of the samples, e.g. C12

The `expData` objects provide different places for different levels of the data: the raw data in the `origin.data` slot; the normalized data in the `norm.data` slot and data after quality control in the `qc.data` slot. In addition, the general quality of each plate is stored in the `plate.quality` slot.

An example of creating a new cellData object is as follows:

```
> (oneCell <- cellData(name = "Average Intensity of Nuclei",
+                       expwell = paste0(rep(LETTERS[2:15],each=20),
+                         rep(formatC(3:22,width=2,flag=0),times=14)),
+                       ctrwell = c("C23","E23","G23")))
```

```
An object of cellData class.
Parameter:  Average.Intensity.of.Nuclei .
Raw Data: NULL.
Data Normalization: ToDo...;
     method: Both Plate Well Ctr Z None.
Quality Control: ToDo....
Significant hits:
      ToDo...
```

4 Data importing and processing

4.1 Raw data importing and re-organization

OperaMate imports all reports of the Columbus analysis system to `expData` objects using the function `loadAll`, and then re-organizes them to several `cellData` objects corresponding to different parameters of the reports. The function `loadAll` requires all reports are of the same data format and are placed in the same location. In addition, all file names should follow the rule: *-replicateID-plateID.txt, e.g. 20130101-s1-03.txt. ('*' should not include the character '-'.)

If the file formats can meet these requirements, assign `cellformat` as "Matrix" or "Tab", and pass the location of the files to `datapath`. Otherwise, you need to assign `cellformat` as "Others", and specify the information of each file by a data.frame variable. See Section 2.1 for its description.

Then, you can import the data and construct `cellData` objects as follows:

```
> ## Data importing
> lstPlates <- loadAll(cellformat="Others",platemap=platemap)
> ## lstPlates <- loadAll(cellformat="Matrix",
> ##                         datapath=dirname(platemap$path[4]))
> ##Data re-organization
> oneCell <- cellLoad( oneCell,lstPlates )
> str(oneCell["origin.data"])
```

```
 num [1:283, 1:12] 114.1 141.4 26.3 81 111.8 ...
 - attr(*, "dimnames")=List of 2
  ..$ : chr [1:283] "B03" "B04" "B05" "B06" ...
  ..$ : Named chr [1:12] "DSIMGA01-s1" "DSIMGA01-s2" "DSIMGA01-s3" "DSIMGA02-s1" ...
  .. ..- attr(*, "names")= chr [1:12] "10" "11" "12" "1" ...
```

If the reports are of other formats, you can redefine the function `parseTemplete`. See the manual for more information.

4.2 Data normalization

Data normalization is to reducing the systematical technical variance among the raw data. Technical sources of variation are unavoidable during experiments, and different amounts of variance occur in different rounds of experiments, which are referred as "batch effect". As to the plate-based assays, different techincal variations are added to different plates, and different locations of well are suffered with different amounts of noise, especially the edge wells. The latter phenomenon is called "edge effects".

Good normalization methods help to attenuate the batch effects. The *OperaMate* package provides several normalization methods including "Ctr", "Plate","Both","Z". The "Ctr" method divides data by the mean of the plate controls. This approach is often favored by biologists. However, as to the large sample screening, it is usually not as accurate as methods which take all samples into consideration. All of the other methods consider all samples, and use the majority of samples as a negative reference. The "Plate" method divides data by the median of their corresponding plates. "Both" employs the Tukey's median polish procedure, and divides data by the median of their corresponding plates and wells recursively. "Z" is the robust z-score method which firstly substracts data by the median of their plates, and then divides them by the median absolute deviation of the plates. The default normalization method is "Both". However, if no normalization methods are expected, you can assign the `norm.method` as "None".

An example is as follows:

```
> oneCell <- cellNorm(oneCell, norm.method="Both")
> str(oneCell["norm.data"])
```

```
 num [1:283, 1:12] 0.905 1.196 0.295 0.797 1.123 ...
 - attr(*, "dimnames")=List of 2
  ..$ : chr [1:283] "B03" "B04" "B05" "B06" ...
  ..$ : Named chr [1:12] "DSIMGA01-s1" "DSIMGA01-s2" "DSIMGA01-s3" "DSIMGA02-s1" ...
  .. ..- attr(*, "names")= chr [1:12] "10" "11" "12" "1" ...
```

Contrasting colors are very useful to visualize the batch effects. The *OperaMate* package provides three ways, the "pricipal component analysis (PCA), hierarchical clustering (heatmap), the scatter plot of the medians. Heatmap method performs hieratical clustering to data matrix, and a large region of distinguishing color indicates the corresponding data with different technical variations. PCA represents data as 2-dimension points based on their two pricipal components. Points distance from others indicate data with excessive noises. Median method plots the median of plates and well positions. Comparisons before and after normalization will be shown side by side if the normalization process has been done. Examples are shown as shown in Figure 1.

```
> cellViz(oneCell,exps="exp",type="heatmap",outpath=outpath,gDevice="png")
> cellViz(oneCell,exps="exp",type="PCA",outpath=outpath,gDevice="png",width=960,height=960)
> cellViz(oneCell,exps="exp",type="median",outpath=outpath,gDevice="png",width=960,height=960)
```

In addition, you can check a specific plate by `plateViz`.

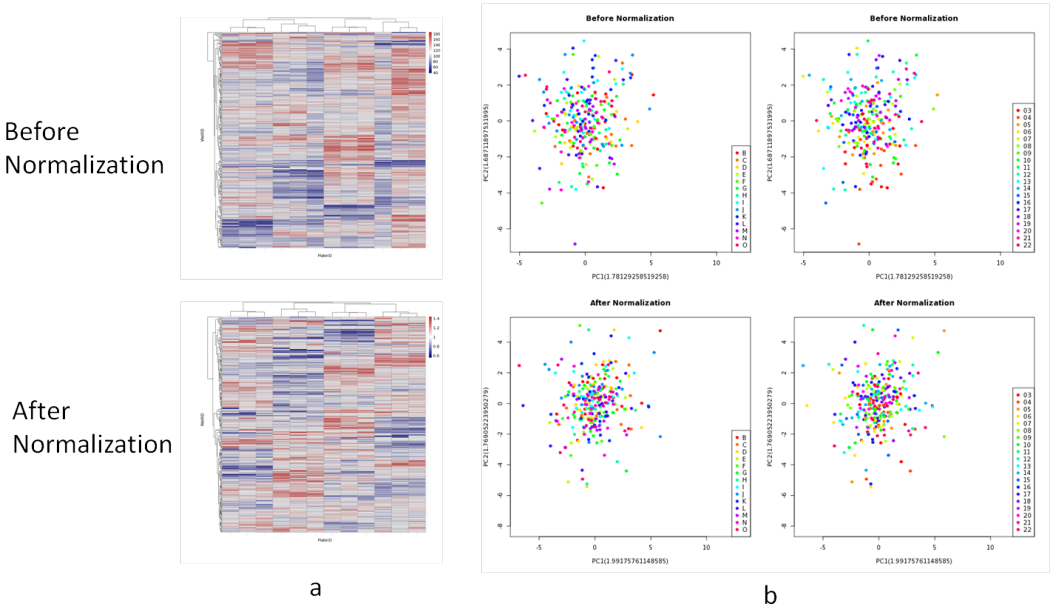

Figure 1: heatmap method (b) PCA method.

```
> plateViz(oneCell,ID="DSIMGA04-s2",gDevice="png",outpath=outpath,width=960,height=960)
```

4.3 Quality control

Before using the data for biological analysis, you should verify that your data passes quality control checks. `OperaMate` provides several ways to do quality control. Firstly, it checks if the duplicated plates have similar distributions. Student t tests are performed between every pair of duplicated plates, and the low quality plates can be found which have small p-values (less than qth, default: 0.05) comparing to others. The data of these plates are replaced by the mean of their duplicates. The qualities of the plates are stored in the `plate.quality` slot. Next, it performs the quality control well by well. We fit a distribution between the SDs and mean values based on all data. If the duplicated wells have SDs much larger than the fitted ones considering the means, they are considered to have low quality. The fitted distribution of SD to mean and the density distribution of the differences between the SDs and their fitted ones can be visualized in this step, as shown in Figure 2. The visualization can be disabled by assigning gDevice="None". An example is as follows:

```
> oneCell <- cellQC( oneCell, qth = .05,
+                    gDevice="png",outpath=outpath,width=960,height=960)
> str(oneCell["qc.data"])
```

```
'data.frame':        1132 obs. of  6 variables:
 $ s1       : num  0.905 1.196 0.295 0.797 1.123 ...
 $ s2       : num  0.833 0.984 0.292 0.733 0.958 ...
 $ s3       : num  0.926 1.093 0.48 0.826 1.12 ...
 $ mean     : num  0.888 1.091 0.356 0.785 1.067 ...
 $ pval     : num  0.807 0.183 0 0.845 0.247 ...
 $ pval.pass: logi  TRUE TRUE FALSE TRUE TRUE TRUE ...
```

```
> head(oneCell["plate.quality"])
```

```
           s1   s2   s3
DSIMGA01 TRUE TRUE TRUE
DSIMGA02 TRUE TRUE TRUE
```

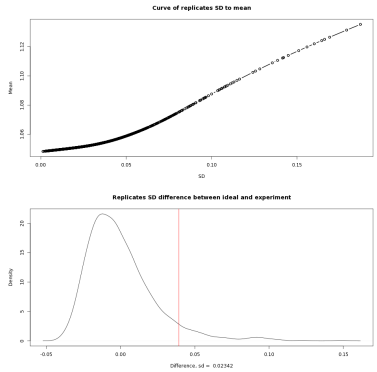

Figure 2: s.d. among the replicated samples. (b) the density distribution of the differences between the s.d.s and the expected ones based on the fitted spline function.

```
DSIMGA03 TRUE TRUE TRUE
DSIMGA04 TRUE TRUE TRUE
```

5 Hit identification and biological analysis

The "hits" are the samples which are meaningfully different from the negative controls. The compulsory method is the t-test between the samples and the controls. The hits are the samples those significantly differ from the negative controls based on t-test (default: p-value less than 0.05). This method is combined with the following methods to reduce a high false positives:

1. `ksd`, $mean \pm k\ standard\,deviation$. The hits are the samples those surpass k (default: 3) standard deviation relative to the mean. This approach is often used with z-score normalization.
2. `kmsd`, $median \pm k\ median\,absolute\,deviation$. The hits are the samples those surpass k (default: 3) median absolute deviation relative to median. It is more robust than `ksd` as to a nonnormal distribution.
3. `ktsd`, t-score method. It is more effective if the data is t-distributed. You can check the fitness to a t distribution by the QQ plot.

An example is as follows:

```
> oneCell <- cellSig(oneCell, method="ktsd",adjust.method="fdr",
+                    gDevice="png",outpath=outpath,width=960,height=960)
> names(oneCell["Sig"])
```

```
[1] "SigMat"    "threshold" "stats"     "pvalue"
```

In addition, the signicance of the hits can be exihibited by a volvano plot,as shown in Figure 3(a).

```
> labels <- c("Axin1")
> names(labels) <- c("DSIMGA04:C07")
> cellSigplot(oneCell,gDevice="png",outpath=outpath,
+             highlight.label=labels,width=960,height=960)
```

Moreover, the potential biological meanings can be interpreted with the help of DAVID Webservice. You are required to register at http://david.abcc.ncifcrf.gov/webservice/register.htm and provide the registered email address the to function. The description of genemap is referred in Section 2.1.

```
> chart <- cellSigAnalysis(oneCell,genemap,
+                          email="cliu@sjtu.edu.cn", type = "all",
+                          david.terms = list(GO=c("GOTERM_BP_FAT","GOTERM_MF_FAT",
+                                                  "GOTERM_CC_FAT")),
```

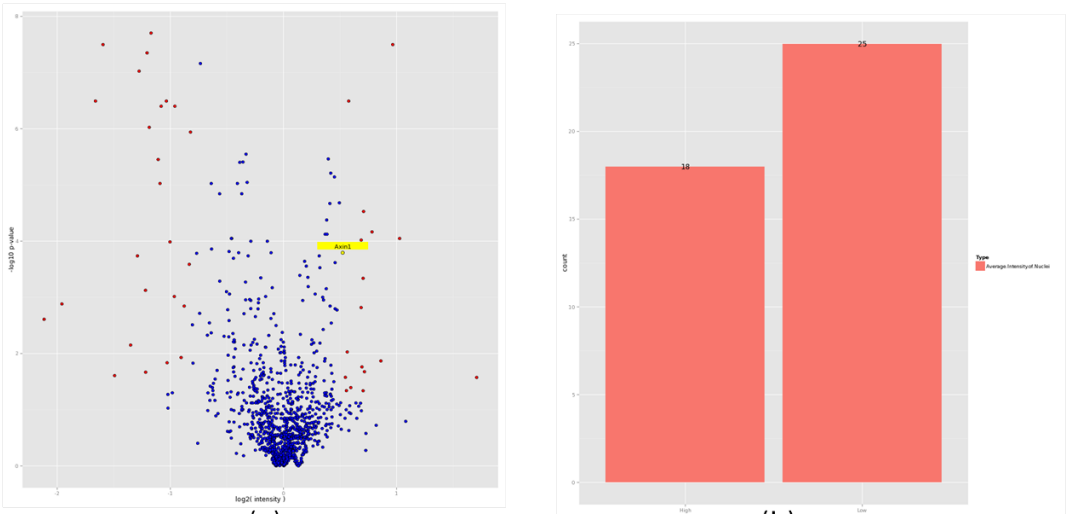

Figure 3: (a) Volcano plot between the log2 intensity and the log10 p-value in the multiple t-tests. The red points are hits statistically and quantitatively significantly different from the negative controls. (b) The counts of the detected hits.

```
+                          file = file.path(outpath,"DAVIDReports.txt"),
+                          gDevice="png",outpath=outpath,width=960,height=960)
> colnames(chart)
```

```
NULL
```

6 Summarization

At last, all processed data and the significant hits of all data type are summarized to a single report, which is easy to check using Microsoft Office Excel or other softwares. The numbers of hits are visualized by a histogram, as shown in Figure 3(b).

```
> anno.data <- GenerateReport(list(oneCell),genemap,
+                             file=file.path(outpath,"OperaMateReports.txt"),
+                             gDevice="png",outpath=outpath,width=960,height=960)
> colnames(anno.data)
```

```
 [1] "Barcode"                                "Well"
 [3] "PoolCatalogNumber"                      "GeneSymbol"
 [5] "GENE.ID"                                "Accession.Number"
 [7] "GI.Number"                              "Average.Intensity.of.Nuclei.s1"
 [9] "Average.Intensity.of.Nuclei.s2"         "Average.Intensity.of.Nuclei.s3"
[11] "Average.Intensity.of.Nuclei.mean"       "Average.Intensity.of.Nuclei.pval"
[13] "Average.Intensity.of.Nuclei.pval.pass" "Average.Intensity.of.Nuclei.Low"
[15] "Average.Intensity.of.Nuclei.High"
```

7 Acknowledgement

We thank Li Mao and his advisor Lin Li for providing the original screening data generated by the Columbus system. The example data in the package are synthesis data generated based on their providing data.

8 Session info

```
R version 3.2.2 Patched (2015-10-08 r69496)
Platform: x86_64-apple-darwin10.8.0 (64-bit)
Running under: OS X 10.6.8 (Snow Leopard)

locale:
[1] en_US.UTF-8/en_US.UTF-8/en_US.UTF-8/C/en_US.UTF-8/en_US.UTF-8

attached base packages:
[1] stats     graphics  grDevices utils     datasets  methods   base

other attached packages:
[1] OperaMate_1.2.0 RSQLite_1.0.0   DBI_0.3.1

loaded via a namespace (and not attached):
 [1] Rcpp_0.12.1             RColorBrewer_1.1-2      plyr_1.8.3
 [4] tools_3.2.2             digest_0.6.8            lattice_0.20-33
 [7] annotate_1.48.0         RDAVIDWebService_1.8.0 gtable_0.1.2
[10] Matrix_1.2-2            graph_1.48.0            Category_2.36.0
[13] parallel_3.2.2          proto_0.3-10            rJava_0.9-7
[16] genefilter_1.52.0       stringr_1.0.0           GOstats_2.36.0
[19] S4Vectors_0.8.0         IRanges_2.4.0           stats4_3.2.2
[22] grid_3.2.2              GSEABase_1.32.0         Biobase_2.30.0
[25] AnnotationDbi_1.32.0    survival_2.38-3         XML_3.98-1.3
[28] RBGL_1.46.0             pheatmap_1.0.7          GO.db_3.2.2
[31] ggplot2_1.0.1           reshape2_1.4.1          magrittr_1.5
[34] splines_3.2.2           scales_0.3.0            MASS_7.3-44
[37] BiocGenerics_0.16.0     AnnotationForge_1.12.0 BiocStyle_1.8.0
[40] colorspace_1.2-6        xtable_1.7-4            labeling_0.3
[43] stringi_0.5-5           munsell_0.4.2
```

References

[1] Michael Boutros, Ligia Bras, and Wolfgang Huber. Analysis of cell-based rnai screens. *Genome Biology*, 7(7):R66, 2006. URL: http://genomebiology.com/2006/7/7/R66, doi:10.1186/gb-2006-7-7-r66.

[2] R Core Team. *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, Vienna, Austria, 2014. URL: http://www.R-project.org/.

[3] Xiaoli Jiao, Brad T. Sherman, Da Wei Huang, Robert Stephens, Michael W. Baseler, H. Clifford Lane, and Richard A. Lempicki. David-ws: a stateful web service to facilitate gene/protein list analysis. *Bioinformatics*, 28(13):1805–1806, 2012. URL: http://bioinformatics.oxfordjournals.org/content/28/13/1805.abstract, arXiv:http://bioinformatics.oxfordjournals.org/content/28/13/1805.full.pdf+html, doi:10.1093/bioinformatics/bts251.