F. Picard, S. Robin, M. Lavielle *et al.* A statistical approach for array CGH data analysis. *BMC Bioinformatics*, 6(1):27, 2005.

E. E. Schadt, S. W. Edwards, D. GuhaThakurta *et al.* A comprehensive transcript index of the human genome generated using microarrays and computational approaches. *Genome Biol*, 5(10):R73, 2004.

V. Stolc, M. Samanta, W. Tongprasit *et al.* Identification of transcribed sequences in Arabidopsis thaliana by using high-resolution genome tiling arrays. *Proc Natl Acad Sci USA*, 102(12):4453–4458, 2005.

L. V. Sun, L. Chen, F. Greil *et al.* Protein-DNA interaction mapping using genomic tiling path microarrays in Drosophila. *Proc Natl Acad Sci USA*, 100:9428–9433, 2003.

T. Toyoda and K. Shinozaki. Tiling array-driven elucidation of transcriptional structures based on maximum-likelihood and Markov models. *Plant J*, 43(4):611–621, 2005.

A. Zeileis, F. Leisch, K. Hornik and C. Kleiber. strucchange: An R package for testing for structural change in linear regression models. *Journal of Statistical Software*, 7:1–38, 2002.

Matt Ritchie
European Bioinformatics Insitute (EBI)
European Molecular Biology Laboratory (EMBL) Cambridge, UK
ritchie@ebi.ac.uk

Wolfgang Huber
European Bioinformatics Insitute (EBI)
European Molecular Biology Laboratory (EMBL) Cambridge, UK
huber@ebi.ac.uk

Analyzing Flow Cytometry Data with Bioconductor

by Nolwenn Le Meur and Florian Hahne

Introduction

In the recent past, flow cytometry (FCM) has become a high-throughput technique used in both basic and clinical research. Applications range from studies focusing on the immunological status of patients, therapeutic approaches involving stem cells up to functional screens used to identify specific phenotypes. The technology is capable of measuring multiple fluorescence as well as some morphological properties of individual cells in a cell population on the basis of light emission. FCM experiments can be extremely complex to analyze due to the large volume of data that is typically created in several processing steps. As an example, flow cytometry high content screening (FC-HCS) can process at a single workstation up to a thousand samples per day each containing thousands of cells, monitoring up to eighteen parameters per sample. Thus, the amount of information generated by these technologies must be stored and managed and finally needs to be summarized in order to make it accessible to the researcher.

Instrument manufacturers have developed software to drive the data acquisition process of their cytometers, but these tools are primarily designed for their proprietary instrument interface and offer few or no high level data processing functions. The packages **rflowcyt** and **prada** provide facilities for importing, storing, assessing and preprocessing data from FCM experiments. In this article we demonstrate the use of these packages for some common tasks in flow cytometry data analysis.

FCS format

In order to facilitate data exchange across different platforms, a data standard has been developed which is now widely accepted by the flow cytometry community and also by most instrument manufacturers. Flow Cytometry Standard (FCS) binary files contain both raw data and accompanying meta data of individual cytometry measurements and optionally the results of prior analyses carried out on the raw data (Seamer et al., 1997). The current version of the FCS standard is 3.0, but both packages can also deal with the old 2.0 standard which is still widely used. We can import FCS files into R using the function `read.fcs`.

Data models

Both **rflowcyt** and **prada** use their own object models to deal with FCM data. While the focus of **rflowcyt** is more on single cytometry measurements, **prada** offers the possibility to combine several individual measurements in the confines of a single experiment and to include all the necessary metadata. Its object model tries to stay close to the familiar micro-array data structures (`expressionSet`) making use of already defined generic functions. Both models store the data corresponding to the different immunofluorescence measurements or variables and the metadata included in the FCS files. The 2 main slots provide:

- a data frame with rows corresponding to the biological unit (i.e. cells) and columns corresponding to the measured variables
- the experimental metadata as a list (**rflowcyt**) or a named vector (**prada**)

The argument `objectModel` to `read.fcs` can be used to chose between the two models when importing the data. In addition, the package **rflowcyt** provides functions for the conversion between objects of both classes.

prada data model

Objects of class `cytoFrame` are the containers for storing individual cytometry measurements in **prada**. The data slot can be accessed using the function `exprs`, the metadata slot via the function `description`. Subsetting of the data is possible using the usual syntax for data frames and matrices.

```
> library(prada)
> data(cframe)
> cframe

cytoFrame object with 2115 cells and 8 observables:
FSC-H SSC-H FL1-H FL2-H FL3-H FL2-A FL4-H Time
slot 'description' has 148 elements

> subset <- cframe[1:3, c(1, 2,
+     3, 7, 8)]
> exprs(subset)

     FSC-H SSC-H FL1-H FL4-H Time
[1,]   467   532    87   449    2
[2,]   437   431    28   478    2
[3,]   410   214     0   358    2

> description(cframe)[4:6]

                              $SYS
"Macintosh System Software 9.2.2"
                           CREATOR
           "CellQuest Pro  4.0.2"
                              $TOT
                            "2115"
```

Collections of several cytometry measurements (whole experiments) can be stored in objects of class `cytoSet`. The phenoData slot of these objects contains all the relevant experiment-wide meta data. Multiple FCS files can be imported along with their metadata when the first argument to `read.fcs` is a vector of filenames or an object of class `phenoData` (see the documentation to `read.fcs` for more details). Subsetting of `cytoSets` is similar to subsetting of list, i.e., individual `cytoFrame` objects are returned when subsetting is done with double brackets.

```
> data(cset)
> cset

cytoSet object with 5 cytoFrames and colnames
 FSC-H SSC-H FL1-H FL2-H FL3-H FL2-A FL4-H Time

> subset <- cset[1:2]
> pData(subset)

                            name  ORF
2 fas-Bcl2-plate323-04-04.A02 MOCK
3 fas-Bcl2-plate323-04-04.A03  YFP
  batch
2     1
3     1

> class(cset[[3]])

[1] "cytoFrame"
attr(,"package")
[1] "prada"
```

`csApply` can be used to apply a function on all items of a `cytoSet`. In a simple case this could for instance be a preprocessing step or a statistical inference on the data from each well. In a more complex application, the function could summarize different features of the data and even produce diagnostic plots for visualization and quality assesment. Here, we apply a preprocessing function which removes artefactual measurements from our dataset based on the morphological properties of a cell and computes the number of cells in each of the wells on the plate.

```
> myFun <- function(xraw) {
+     fn <- fitNorm2(xraw[, c("FSC-H",
+         "SSC-H")], scale = 2)
+     x <- xraw[fn$sel, ]
+     return(nrow(x))
+ }
> cellCounts <- csApply(cset,
+     myFun)
```

Many of the common R methods like `plot` or `length` are also available for objects of class `cytoFrame` and `cytoSet`.

rflowcyt data model

Objects of class `FCS` are the containers for storing individual cytometry measurements in **rflowcyt**. The data slot can be accessed using the function `fluors`, the metadata slot via the function `metaData`. Subsetting of the data is possible using:

- [i,j] to extract or subset information from the data (a matrix object) of the FCS R-object
- [[i]] to extract metadata (which is of S4 class FCSmetadata) of the FCS R-object

```
> library(rflowcyt)
> data(VRCmin)
> st.DRT

Original Object of class FCS from:
DRT_GAG.fcs
Object name: st.DRT
Dimensions 206149 by 8

> subset <- st.DRT[1:3,1:3]
> metaData(subset)

 FACSmetadata for non-original FCS object:
st.DRT from original file DRT_GAG.fcs
 with  3 cells and 3 parameters.

> fluors(subset)
   FSC-Height Side Scatter CD8 FITC
1         640          458      298
2         136          294      102
3         588          539      265
```

Multiple cytometry measurements can be imported when the filename argument to read.fcs is a vector of file names and are stored as a list of individualFCS objects. These lists may be further processed using the familiar basic R functions, however, no experiment-wide metadata is provided.

Besides the FCS class, **rflowcyt** include FCSmetadata, FCSsummary, and FCSgate classes. FCSmetadata is the class of the metadata slot of an FCS R-object. The FCSsummary class is the class of the output of the summary method implemented on a FCS R-object. The FCSgate class contains the FCS class and extends it to include gating information (for more details, see the following section).

Gating

A common task in the analysis of flow cytometry data is to perform interactive selections of subpopulations of cells with respect to one or several measurement parameters, a process known as gating. In this respect, a gate is a set of rules that uniquely identifies a cell to be part of a given subpopulation. In the easiest case this can be a sharp cutoff, e.g., all cells with values in one parameter that are larger than a given threshold. But often much more complex selections are necessary like rectangular or elliptic areas in two dimensions or even polygonal boundaries. It is sometimes desirable to define a gate on a data set and later on apply this gate to a number of additional data sets, hence gates should be independent from the actual raw data. In addition, there may be several different combinations of gates that can be combined in a logical manner (i.e., "AND" and "OR") and in a defined order, thus the concept of the gate can be extended to collections of multiple gates.

The package **prada** offers the infrastructure to apply gating on cytometry data. Objects of class gate and gateSet model the necessary features of individual gates and of collections of multiple gates and can be assigned to the gate slot of objects of class cytoFrame. Gates can either be created from scratch by specifying the necessary selection rules or, much more conveniently, the function drawGate can be used to interactively set gates based on two-dimensional scatter plots of the raw data (Fig. 1). Please see the vignette of package **prada** for a more thorough discussion on gating.

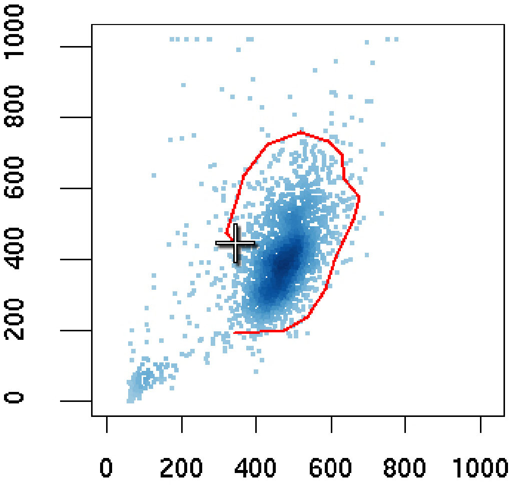

Figure 1: Interactive drawing of a polygonal gate based on a scatterplot of two cytometry parameters using function drawGate.

Quality control and quality assessment of cytometry data

Data quality control and quality assessment are crucial steps in processing high throughput FCM data. Quality control efforts have been made in clinical cell analysis by flow cytometry. For example, guidelines were defined to monitor the fluorescence measurements by computing calibration plots for each fluorescent parameter. However such procedures are not yet systematically applied in high throughput FCM and quality assessment of the raw data is often needed to overcome the lack of data quality control. The aim of data quality assessment is to detect systematic and stochastic effects that are not likely to be biologically motivated. The rationale is that systematic errors often indicate the need for adjustments in sample handling or processing. Further, the aberrant samples should be identified and potentially removed from any downstream analysis in order to avoid spurious results.

rflowcyt proposes a variety of graphical exploratory data analytic (EDA) tools to explore and summarize ungated FCM data.

- `plotECDF.FCS` creates Empirical Cumulative Distribution (ECDF) plots that reveal differences in distributions (Fig. 2);

- `boxplot.FCS` draws boxplots that display location and variation of the distributions and facilitate the comparison of these features between samples as they are aligned horizontally;

- `plotQA.FCS` summarizes the distribution of one or two parameters by their means, medians, modes or IQR for the diferent samples and displays the values in a scatterplot (Fig. 4). The dots in the resulting scatterplot can be colored according to the samples position in a 96-well plate to reveal potential plate effects;

- `plotdensity.FCS` displays density curves that reveal the shape of the distributions, especially multi-modality and asymmetry;

We illustrate the usefulness of those visualization tools to assess FCM data quality through examination of a collection of weekly peripheral blood samples obtained from a patient following allogeneic blood and marrow transplant. Samples were taken at various time points before and after transplantation. At each time point, every blood sample was divided into eight aliquots. Values for the forward light scatter (FSC) which measures a cell's size and for the sideward light scatter (SSC, a measure for a cell's granularity) of aliquots from the same sample should therefore be comparable.

The `plotECDF.FCS` function can be used to visualize several variables for several samples in the same graph.

```
> data(flowcyt.data)
> subset <- flowcyt.data[c(1:24,
+     41:48, 57:72)]
> stain <- paste("A", 1:8, sep = "")
> timePoint <- c(-8, 0, 5, 27,
+     39, 46)
```

```
> plotECDF.FCS(subset,
+               varpos = c(1),
+              var.list = c(paste("Day ",
+              timePoint)),
+              group.list = stain,
+              type = "l", xlab = "FSC",
+              lwd = 2, cex = 1.5)
```

For example, Figure 2 shows the FSC parameter for the 8 aliquots of a sample per time point. Each panel corresponds to a particular time point, in days before or after transplantation. In Figure 2 we expected to see the density curves superimpose. One aliquot significantly deviates from the other. This aliquot should be investigated in more detail and potentially be removed from further analysis.

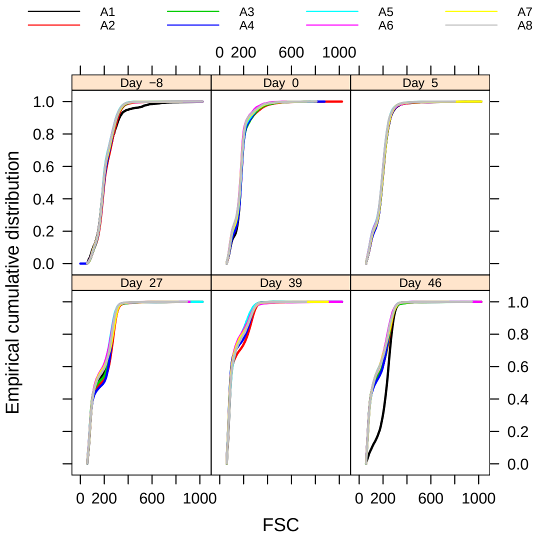

Figure 2: ECDF plots of the FSC parameter for 8 aliquots of a sample at different time point. Each panel corresponds to a particular time point, in days before or after transplantation. In each panel, each intensity curves represents one of the 8 aliquots.

ECDF plots are not good for visualizing the shape of the distributions. Instead, you can use the function `plotdensity.FCS` (Fig. 3).

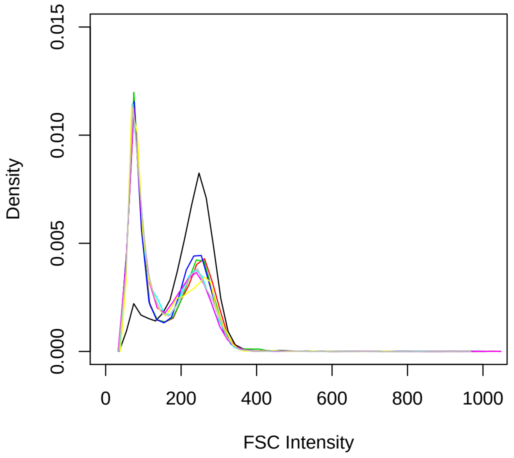

Figure 3: Density plots of the FSC parameter for 8 aliquots of the same sample.

```
> plotdensity.FCS(subset[41:48],
+     varpos = c(1), ylab = "Density",
+     xlab = "FSC Intensity",
+     col = c(1:8), ylim = c(0,
+         0.015))
```

Finally the `plotQA.FCS` function creates "summary" scatterplots to visualize samples relationship within plates. This representation allows to identify biological outlier and/or plate biases, such as edge effect or within-plate spatial effect. Figure 4 shows the SSC *vs* FSC median intensities for all aliquots stored in one 96-well plate and colored by their column position in the plate. In this figure, if all samples were identical, we expect to see a single cluster of data points. One has to be careful when interpreting such plots as each column correspond to different samples collected at different time points. However, we note that some columns have widely spread values (light blue and brown) and that one aliquot is an outlier as it is far away from the rest of its group. This aliquot appears to be the same as in Figures 2 and 3.

```
> idx <- order(names(flowcyt.data))
> flowcyt.data <- flowcyt.data[idx]
> plotQA.FCS(flowcyt.data, varpos = c(1,
+     2), col = "col", median,
+     labeling = TRUE, xlab = "SSC median",
+     ylab = "FSC median", xlim = c(0,
+         200), ylim = c(75, 275),
+     pch = "*", asp = 1, cex = 1.5,
+     main = "")
```

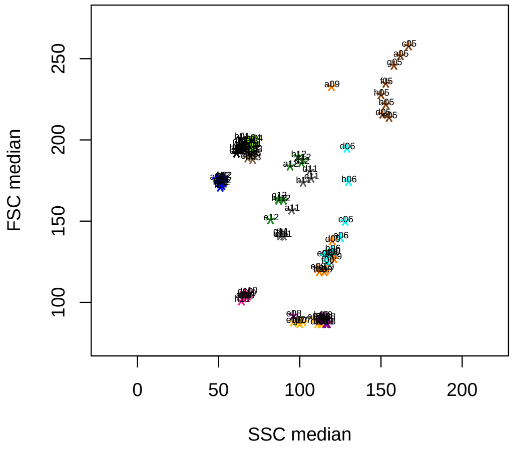

Figure 4: Scatterplot of SSC *vs* FSC medians intensities for one plate.

prada offers another visualization tool which can be used to inspect the data from whole experiments. Using the function `plotPlate` we can display quantitative as well as qualitative values or even complex graphs for each well of a microtiter plate retaining its array format 2. This allows for the identification of spatial effects and for a consise presentation of important features of an experiment. `plotPlate` is implemented using grid graphic and users are able to define their own plotting functions, so conceptually anything can be plotted in a mircrotiter plate format (see Figure 5).

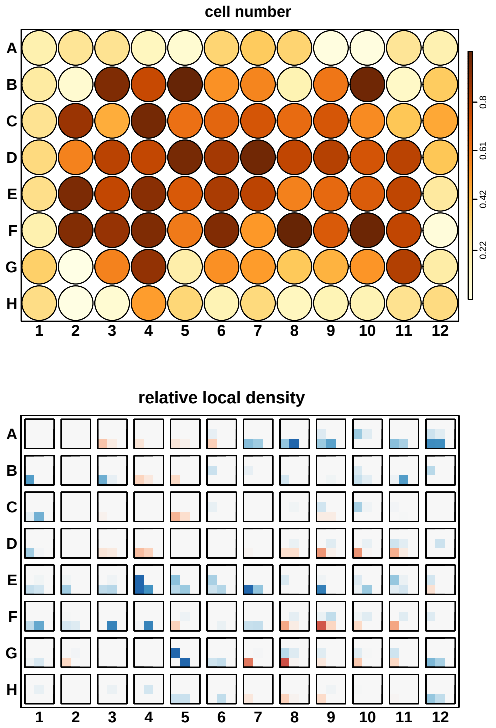

Figure 5: Two different variations of plate plots for 96 well microtiter plates. Top: Quantitative values. The consistently low number of cells around the edges of the plate indicates a technical problem. Bottom: Complex graph. Image maps of two-dimensional local densities of FSC vs SSC values for each well relative to a standard. Blue areas indicate low, red areas indicate high local densities. These plots help detect morphological changes in a cell population.

Discussion and Conclusion

The application of flow cytometry in modern cell biology is diverse and so are the demands on data analysis. The multitude of packages within R and Bioconductor already provides for many tools that are also useful in the analysis of FCM data. The packages **rflowcyt** and **prada** try to close the gap between data acquisition and data analysis by enabling the researches to take their data into the powerful R environment and to make use of the statistical and graphical solutions already available there. In addition, they provide for tools that are commonly used in early steps of data analysis which in principle are the same for all FCM applications.

Currently, in a collaboration of several groups involved in high-throughput FCM together with instrument manufacturers and members of the flow cytometry standards initiative a **flowCore** package and a number of additional FCM utility packages are developed. The aim is to merge both **prada** and **rflowcyt** into one core package which is copmpliant with the data exchange standards that are currently developed in the community. Visualization as well as quality control will than be part of the utility packages that depend on the data structures defined in the **flowCore** package.

Bibliography

L. C. Seamer, C. B. Bagwell, L. Barden *et al.* Proposed new data file standard for flow cytometry, version fcs 3.0. *Cytometry*, 28(2):118–122, Jun 1997.

Nolwenn Le Meur
Computational Biology
Fred Hutchinson Cancer Research Center
Seattle, WA, USA
nlemeur@fhcrc.org

Florian Hahne
Molecular Genome Analysis
German Cancer Research Center
Heidelberg, Germany
f.hahne@dkfz.de

Protein Complex Membership Estimation using apComplex

by Denise Scholtens

Graphs of protein-protein interactions, so called 'interactomes', are rapidly surfacing in the systems biology literature. In these graphs, nodes represent cellular proteins and edges represent interactions between them. Global interactome analyses are often undertaken to explore topological features such as network diameter, clustering coefficients, and node degree distribution. Local interactome modeling, particularly at the protein complex level, is also important for identifying distinct functional components of the cell and studying their interactivity (Hartwell et al., 1999). The **apComplex** package contains functions to locally estimate protein complex membership as described in Scholtens and Gentleman (2004) and Scholtens et al. (2005).

Two technologies are generally used to query protein-protein relationships. Affinity purification-mass spectrometry (AP-MS) technologies detect protein complex co-membership. In these experiments a set of proteins are used as baits, and in separate purifications, each bait identifies all hits with which it shares protein complex membership. AP-MS baits and their hits may physically bind to each other, or they may be joined together in a complex through an intermediary protein or set of proteins. If a bait protein is a member of more than one complex, all of its hits may not necessarily themselves be complex co-members. These biological realities become essential components of complex membership estimation. Publicly available AP-MS data sets for *Saccharomyces cerevisiae* include those reported by Gavin et al. (2002, 2006), Ho et al. (2002), and Krogan et al. (2004, 2006).

Yeast-two-hybrid (Y2H) technology is another bait-hit system that measures direct physical interactions. The distinction between AP-MS and Y2H data is subtle, but crucial. Two proteins that are part of the same complex may not physically interact with each other. Thus an interaction detected by AP-MS may not be detected by Y2H. On the other hand, two proteins that do physically interact by definition form a complex so any interaction detected by Y2H should also be detected by AP-MS. Under the same experimental conditions, Y2H technology should in fact consist of a subset of the interactions detected by AP-MS technology, the subset consisting of complex co-members that are physically bound to each other. Ito et al. (2001) and Uetz et al. (2000) both offer publicly available Y2H data sets for *Saccharomyces cerevisiae*.

apComplex deals strictly with data resulting from AP-MS experiments. The joint analysis of Y2H and AP-MS data is an interesting and important problem and is in fact an obvious next step after complex membership estimation, but is not currently dealt with in **apComplex**.