# Characterization of 786-O and 786-O/VHL Cell-Derived Extracellular Vesicle Subpopulations

# PART I - CD24 MEASUREMENT USING FLOW CYTOMETRY

## Supplementary Information:

Data analysis and annotated source code

Guillaume Pelletier

08 November 2020

**Abstract**

In this analysis, we suggest a "best practices" method using open-source software (R/bioconductor) for analyzing small-particle flow cytometry data in respect to the potential biases attributed to the "swarm effect". A special experimental design is required for analysis of the swarm effect in samples containing small particles, an example of which is described here.

A deconvolution technique is applied for the first time to the analysis of swarm effect on population counts using maximum-likelihood (ML) expectation-maximization (EM) estimates for mixed distribution modeling. Constraints derived from experimental design allows for simplification of distribution parameters. We implement a noise-reduction technique to minimize noise-related artefacts across sample dilutions. We suggest a simple statistical method for interpreting population count differences across samples and across sample dilutions.

The results of significance testing for our data appear to indicate a statistically significant difference between 786-O and VHL cell-derived extracellular vesicles for the absolute count of CD24mid events, as well as for the overall proportion of CD24+ events and the overall proportion of CD24mid events. While multiple comparisons statistical testing reveals no significant differences in CD24hi population counts, fingerprinting reveals that most of the overall between-sample variation occurs in the CD24hi region. Lack of significance is perhaps owing to the difficulty of detecting rare events in very dilute samples. These data also provide with a better understanding of below-threshold particle size differences between samples: in particular, 786-O cells are found to generate significantly more CD24hi EVs smaller than the size detection threshold of the instrument compared to 786-O/VHL EVs.

Special thanks to Pr Luc H Boudreau, Pr Sandra Turcotte, the Atlantic Cancer Research Institute, NBIF and Mr David G Sebolsky for supporting this research.

## Contents

# 1 Production environment

## 1.1 Required libraries

In order to install all of the required libraries listed below, you will first need to source from **bioconductor**. For more information about **bioconductor**, please visit https://www.bioconductor.com/.

When troubleshooting any of the following code, please note that the order of library imports matters, as certain packages may interfere with each other in unpredictable ways.

```
library("Biobase")
library("gridExtra")
library("reshape2")
library("HH")
library("mixtools")
library("diptest")
library("ggplot2")
library("dplyr")
library("tidyr")
library("purrr")
library("car")

# Flow cytometry specific packages
library("flowCore")
library("flowWorkspace")
library("flowUtils")
library("flowStats")
library("flowFP")
library("MetaCyto")
library("ggcyto")

# Used to generate this annotated PDF report
# https://haozhu233.github.io/kableExtra/awesome_table_in_pdf.pdf
library("kableExtra")
library("pander")
library("knitr")
```

## 1.2 SessionInfo()

This may be useful for troubleshooting and is included for the sake of reproducibility.

```
pander::pander(utils::sessionInfo(), locale=TRUE, compact=TRUE)
```

**R version 4.0.2 (2020-06-22)**

**Platform:** x86_64-apple-darwin17.0 (64-bit)

**locale:** en_US.UTF-8||en_US.UTF-8||en_US.UTF-8||C||en_US.UTF-8||en_US.UTF-8

**attached base packages:** *grid*, *parallel*, *stats*, *graphics*, *grDevices*, *utils*, *datasets*, *methods* and *base*

**other attached packages:** *knitr(v.1.30)*, *pander(v.0.6.3)*, *kableExtra(v.1.3.1)*, *ggcyto(v.1.18.0)*, *ncdfFlow(v.2.36.0)*, *BH(v.1.72.0-3)*, *RcppArmadillo(v.0.10.1.0.0)*, *MetaCyto(v.1.12.0)*, *flowFP(v.1.48.0)*, *flowViz(v.1.54.0)*, *flowStats(v.4.2.0)*, *flowUtils(v.1.54.0)*, *flowWorkspace(v.4.2.0)*, *flowCore(v.2.2.0)*, *car(v.3.0-10)*, *carData(v.3.0-4)*, *purrr(v.0.3.4)*, *tidyr(v.1.1.2)*, *dplyr(v.1.0.2)*, *ggplot2(v.3.3.2)*, *diptest(v.0.75-7)*, *mixtools(v.1.2.0)*, *HH(v.3.1-42)*, *multcomp(v.1.4-14)*, *TH.data(v.1.0-10)*, *MASS(v.7.3-53)*, *survival(v.3.2-7)*, *mvtnorm(v.1.1-1)*, *latticeExtra(v.0.6-29)*, *lattice(v.0.20-41)*, *reshape2(v.1.4.4)*, *gridExtra(v.2.3)*, *Biobase(v.2.50.0)* and *BiocGenerics(v.0.36.0)*

**loaded via a namespace (and not attached):** *readxl(v.1.3.1)*, *backports(v.1.2.0)*, *Hmisc(v.4.4-1)*, *wrapr(v.2.0.4)*, *plyr(v.1.8.6)*, *igraph(v.1.2.6)*, *ConsensusClusterPlus(v.1.54.0)*, *splines(v.4.0.2)*, *gmp(v.0.6-1)*, *fda(v.5.1.5.1)*, *digest(v.0.6.27)*, *htmltools(v.0.5.0)*, *fansi(v.0.4.1)*, *magrittr(v.1.5)*, *checkmate(v.2.0.0)*, *CytoML(v.2.2.1)*, *cluster(v.2.1.0)*, *ks(v.1.11.7)*, *aws.signature(v.0.6.0)*, *openxlsx(v.4.2.3)*, *fastcluster(v.1.1.25)*, *RcppParallel(v.5.0.2)*, *matrixStats(v.0.57.0)*, *sandwich(v.3.0-0)*, *cytolib(v.2.2.0)*, *jpeg(v.0.1-8.1)*, *colorspace(v.1.4-1)*, *rvest(v.0.3.6)*, *rrcov(v.1.5-5)*, *haven(v.2.3.1)*, *xfun(v.0.19)*, *crayon(v.1.3.4)*, *jsonlite(v.1.7.1)*, *hexbin(v.1.28.1)*, *graph(v.1.68.0)*, *zoo(v.1.8-8)*, *glue(v.1.4.2)*, *gtable(v.0.3.0)*, *zlibbioc(v.1.36.0)*, *webshot(v.0.5.2)*, *kernlab(v.0.9-29)*, *IDPmisc(v.1.1.20)*, *Rgraphviz(v.2.34.0)*, *Rmpfr(v.0.8-1)*, *DEoptimR(v.1.0-8)*, *abind(v.1.4-5)*, *scales(v.1.1.1)*, *Rcpp(v.1.0.5)*, *viridisLite(v.0.3.0)*, *xtable(v.1.8-4)*, *htmlTable(v.2.1.0)*, *foreign(v.0.8-80)*, *mclust(v.5.4.6)*, *FlowSOM(v.1.22.0)*, *Formula(v.1.2-4)*, *stats4(v.4.0.2)*, *tsne(v.0.1-3)*, *vcd(v.1.4-8)*, *htmlwidgets(v.1.5.2)*, *httr(v.1.4.2)*, *RColorBrewer(v.1.1-2)*, *ellipsis(v.0.3.1)*, *farver(v.2.0.3)*, *pkgconfig(v.2.0.3)*, *XML(v.3.99-0.5)*, *nnet(v.7.3-14)*, *labeling(v.0.4.2)*, *tidyselect(v.1.1.0)*, *rlang(v.0.4.8)*, *later(v.1.1.0.1)*, *munsell(v.0.5.0)*, *cellranger(v.1.1.0)*, *tools(v.4.0.2)*, *cli(v.2.1.0)*, *generics(v.0.1.0)*, *aws.s3(v.0.3.21)*, *evaluate(v.0.14)*, *stringr(v.1.4.0)*, *fastmap(v.1.0.1)*, *yaml(v.2.2.1)*, *zip(v.2.1.1)*, *robustbase(v.0.93-6)*, *nlme(v.3.1-150)*, *RBGL(v.1.66.0)*, *mime(v.0.9)*, *leaps(v.3.1)*, *xml2(v.1.3.2)*, *compiler(v.4.0.2)*, *rstudioapi(v.0.11)*, *curl(v.4.3)*, *png(v.0.1-7)*, *tibble(v.3.0.4)*, *pcaPP(v.1.9-73)*, *stringi(v.1.5.3)*, *forcats(v.0.5.0)*, *Matrix(v.1.2-18)*, *vctrs(v.0.3.4)*, *pillar(v.1.4.6)*, *lifecycle(v.0.2.0)*, *RUnit(v.0.4.32)*, *lmtest(v.0.9-38)*, *data.table(v.1.13.2)*, *corpcor(v.1.6.9)*, *httpuv(v.1.5.4)*, *R6(v.2.5.0)*, *promises(v.1.1.1)*, *KernSmooth(v.2.23-18)*, *RProtoBufLib(v.2.2.0)*, *rio(v.0.5.16)*, *codetools(v.0.2-18)*, *assertthat(v.0.2.1)*, *withr(v.2.3.0)*, *metafor(v.2.4-0)*, *S4Vectors(v.0.28.0)*, *hms(v.0.5.3)*, *rpart(v.4.1-15)*, *rmarkdown(v.2.5)*, *segmented(v.1.3-0)*, *shiny(v.1.5.0)* and *base64enc(v.0.1-3)*

## 1.3 Global variables and convenience functions

```
# This is where all my data files reside on my local machine
setwd("/Users/gp/Documents/Maitrise/Results/N3L2/CD24 quantification/")

# The following colours are used consistently enough in this document
colorPalette <- list(
  "red"   = "#E87D72",
  "blue"  = "#54BCC2",
  "green" = "#00AA00"
)


# Output dataframe in PDF-friendly format
printDataFrame <- function(dataframe, caption, fontsize=7) {
  knitr::kable(
    dataframe,
    format="latex",
    booktabs=TRUE,
    longtable=TRUE,
    caption=caption
  ) %>%
  kableExtra::kable_styling(
    font_size=fontsize,
    latex_options=c(
      "hold position",
      "repeat_header",
      "striped"
    )
  )
}

# kmeans() returns an error when one of the clusters turns out empty. Since in
# context we expect this to happen, we can define more useful behaviour.
safeClustering <- function(
  data,
  centers=matrix(data=c(0, 0, 0, 0), ncol=2),
  principal=1
) {
  # Linter bindings
  cluster <- NULL

  attemptClustering <- try(
    expr={
      km <- stats::kmeans(data, centers=centers)$cluster
    },
    silent=TRUE
  )

  if (class(attemptClustering) == "try-error") {
    km <- principal
  }

  km
```

```
}

# This function takes an anova object, the return value of aov(), and returns
# the p-Value of row 1 (ie.: [[1]])
getANOVApValue <- function(aov) {
  summary(aov)[["Error: Within"]][[1]][["Pr(>F)"]][1]
}
```

# 2 Data pre-processing

## 2.1 Experiment meta-data

The **Biobase** package provides a standard class called `AnnotatedDataFrame` to store and manipulate the experiment metadata.

**References:**

- https://www.ncbi.nlm.nih.gov/pmc/articles/PMC4509590/

**Technical documentation:**

- https://bioconductor.org/packages/devel/bioc/vignettes/Biobase/inst/doc/ExpressionSetIntroduction.pdf
- https://bioconductor.org/packages/devel/bioc/manuals/Biobase/man/Biobase.pdf

Experiment metadata is stored in a tab-delimited `.txt` file, which can be read using `read.table()`. In this case, as the FC500 instrument does not support integration with any of the **bioconductor** tools, the `phenoData.txt` file was created by hand in **Microsoft Excel**. Alternatively, any other plain text editor may be used. Let's take a look at the contents of this file:

```
pData <- utils::read.table(
  file="phenoData.txt",
  header=TRUE,
  row.names=1,
  sep="\t",
  as.is=TRUE
)

printDataFrame(
  pData,
  "phenoData.txt",
  fontsize=7
)
```

Table 1: phenoData.txt

| | id | cellLine | n | a23187 | dilution | facetRow | facetCol | temps | | |
|---|---|---|---|---|---|---|---|---|---|---|
| data/786 n1/786n1_Ca1uM4h_samp_20x 00016119 613.LMD | 1 | 786-O | 1 | 1 | 20 | r1 | c1 | 4 | | |
| data/786 n1/786n1_Ca1uM4h_samp_60x 00016122 616.LMD | 2 | 786-O | 1 | 1 | 60 | r1 | c2 | 4 | | |
| data/786 n1/786n1_Ca1uM4h_samp_100x 00016125 619.LMD | 3 | 786-O | 1 | 1 | 100 | r1 | c3 | 4 | | |
| data/786 n1/786n1_Ca1uM4h_samp_140x 00016128 622.LMD | 4 | 786-O | 1 | 1 | 140 | r1 | c4 | 4 | | |
| data/786 n1/786n1_Ca1uM4h_samp_180x 00016131 625.LMD | 5 | 786-O | 1 | 1 | 180 | r1 | c5 | 4 | | |
| data/786 n1/786n1_Ca1uM4h_samp_220x 00016134 628.LMD | 6 | 786-O | 1 | 1 | 220 | r1 | c6 | 4 | | |
| data/786 n2/786n2_Ca1uM4h_samp_20x 00016158 632.LMD | 7 | 786-O | 2 | 1 | 20 | r2 | c1 | 4 | | |
| data/786 n2/786n2_Ca1uM4h_samp_60x 00016161 635.LMD | 8 | 786-O | 2 | 1 | 60 | r2 | c2 | 4 | | |
| data/786 n2/786n2_Ca1uM4h_samp_100x 00016164 638.LMD | 9 | 786-O | 2 | 1 | 100 | r2 | c3 | 4 | | |
| data/786 n2/786n2_Ca1uM4h_samp_140x 00016167 641.LMD | 10 | 786-O | 2 | 1 | 140 | r2 | c4 | 4 | | |
| data/786 n2/786n2_Ca1uM4h_samp_180x 00016170 644.LMD | 11 | 786-O | 2 | 1 | 180 | r2 | c5 | 4 | | |
| data/786 n2/786n2_Ca1uM4h_samp_220x 00016173 647.LMD | 12 | 786-O | 2 | 1 | 220 | r2 | c6 | 4 | | |
| data/786 n3/786n3_Ca1uM4h_samp_20x 00016176 650.LMD | 13 | 786-O | 3 | 1 | 20 | r3 | c1 | 4 | | |
| data/786 n3/786n3_Ca1uM4h_samp_60x 00016179 653.LMD | 14 | 786-O | 3 | 1 | 60 | r3 | c2 | 4 | | |
| data/786 n3/786n3_Ca1uM4h_samp_100x 00016182 656.LMD | 15 | 786-O | 3 | 1 | 100 | r3 | c3 | 4 | | |
| data/786 n3/786n3_Ca1uM4h_samp_140x 00016185 659.LMD | 16 | 786-O | 3 | 1 | 140 | r3 | c4 | 4 | | |
| data/786 n3/786n3_Ca1uM4h_samp_180x 00016188 662.LMD | 17 | 786-O | 3 | 1 | 180 | r3 | c5 | 4 | | |
| data/786 n3/786n3_Ca1uM4h_samp_220x 00016191 665.LMD | 18 | 786-O | 3 | 1 | 220 | r3 | c6 | 4 | | |
| data/VHL n1/VHLn1_Ca1uM4h_samp_20x 00016196 670.LMD | 19 | VHL | 1 | 1 | 20 | r4 | c1 | 4 | | |
| data/VHL n1/VHLn1_Ca1uM4h_samp_60x 00016199 673.LMD | 20 | VHL | 1 | 1 | 60 | r4 | c2 | 4 | | |
| data/VHL n1/VHLn1_Ca1uM4h_samp_100x 00016202 676.LMD | 21 | VHL | 1 | 1 | 100 | r4 | c3 | 4 | | |
| data/VHL n1/VHLn1_Ca1uM4h_samp_140x 00016205 679.LMD | 22 | VHL | 1 | 1 | 140 | r4 | c4 | 4 | | |
| data/VHL n1/VHLn1_Ca1uM4h_samp_180x 00016208 682.LMD | 23 | VHL | 1 | 1 | 180 | r4 | c5 | 4 | | |
| data/VHL n1/VHLn1_Ca1uM4h_samp_220x 00016211 685.LMD | 24 | VHL | 1 | 1 | 220 | r4 | c6 | 4 | | |

Table 1: phenoData.txt *(continued)*

| | id | cellLine | n | a23187 | dilution | facetRow | facetCol | temps | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| data/VHL n2/VHLn2_Ca1uM4h_samp_20x 00016215 689.LMD | | | 25 | VHL | 2 | 1 | 20 | r5 | c1 | 4 | |
| data/VHL n2/VHLn2_Ca1uM4h_samp_60x 00016218 692.LMD | | | 26 | VHL | 2 | 1 | 60 | r5 | c2 | 4 | |
| data/VHL n2/VHLn2_Ca1uM4h_samp_100x 00016221 695.LMD | | | 27 | VHL | 2 | 1 | 100 | r5 | c3 | 4 | |
| data/VHL n2/VHLn2_Ca1uM4h_samp_140x 00016224 698.LMD | | | 28 | VHL | 2 | 1 | 140 | r5 | c4 | 4 | |
| data/VHL n2/VHLn2_Ca1uM4h_samp_180x 00016227 701.LMD | | | 29 | VHL | 2 | 1 | 180 | r5 | c5 | 4 | |
| data/VHL n2/VHLn2_Ca1uM4h_samp_220x 00016230 704.LMD | | | 30 | VHL | 2 | 1 | 220 | r5 | c6 | 4 | |
| data/VHL n3/VHLn3_Ca1uM4h_samp_20x 00016234 708.LMD | | | 31 | VHL | 3 | 1 | 20 | r6 | c1 | 4 | |
| data/VHL n3/VHLn3_Ca1uM4h_samp_60x 00016237 711.LMD | | | 32 | VHL | 3 | 1 | 60 | r6 | c2 | 4 | |
| data/VHL n3/VHLn3_Ca1uM4h_samp_100x 00016240 714.LMD | | | 33 | VHL | 3 | 1 | 100 | r6 | c3 | 4 | |
| data/VHL n3/VHLn3_Ca1uM4h_samp_140x 00016243 717.LMD | | | 34 | VHL | 3 | 1 | 140 | r6 | c4 | 4 | |
| data/VHL n3/VHLn3_Ca1uM4h_samp_180x 00016246 720.LMD | | | 35 | VHL | 3 | 1 | 180 | r6 | c5 | 4 | |
| data/VHL n3/VHLn3_Ca1uM4h_samp_220x 00016249 723.LMD | | | 36 | VHL | 3 | 1 | 220 | r6 | c6 | 4 | |

The row labels correspond to the path of all the sample data files (relative to `getwd()`, as defined previously with `setwd()`). There are two cell lines in this experiment: the ATCC 786-O renal cell carcinoma cell line, as well as a 786-O/VHL line with functional VHL gene. We can see that this experiment features biological triplicates, and that each sample is measured in serial dilutions of 1:20, 1:60, 1:100, 1:140, 1:180 and 1:220. The experimental conditions corresponding to each sample file are otherwise identical (adherent cells are incubated with 1 uM A23187 in serum-starved medium for 4 hours before conditioned medium is collected, processed and analyzed).

The variables `facetRow` and `facetCol` have been added for convenience, as these data will be organized in an ordered grid, much like a plate. Let's label our experiment columns and rows while we're at it: these labels will be used for plotting later on.

```
# To make the faceted graph easier to interpret we can label columns and rows
facetLabels <- c(
  "c1"    = "1:20",
  "20"    = "1:20",
  "c2"    = "1:60",
  "60"    = "1:60",
  "c3"    = "1:100",
  "100"   = "1:100",
  "c4"    = "1:140",
  "140"   = "1:140",
  "c5"    = "1:180",
  "180"   = "1:180",
  "c6"    = "1:220",
  "220"   = "1:220",
  "r1"    = "786 n1",
  "r2"    = "786 n2",
  "r3"    = "786 n3",
  "786-0" = "786-0 (n=3)",
  "r4"    = "VHL n1",
  "r5"    = "VHL n2",
  "r6"    = "VHL n3",
  "VHL"   = "786-0/VHL (n=3)"
)
```

The `AnnotatedDataFrame` is then created directly from `pData`, an object of type `DataFrame`.

```
# Otherwise flowCore::read.flowSet() doesn't work
pData$filename <- row.names(pData)

phenoData <- new(
```

```
  "AnnotatedDataFrame",
  data=pData,
  varMetadata=data.frame(
    labelDescription=BiocGenerics::colnames(pData),
    row.names=BiocGenerics::colnames(pData)
  )
)

phenoData
```

```
## An object of class 'AnnotatedDataFrame'
##   rowNames: data/786 n1/786n1_Ca1uM4h_samp_20x 00016119 613.LMD
##     data/786 n1/786n1_Ca1uM4h_samp_60x 00016122 616.LMD ... data/VHL
##     n3/VHLn3_Ca1uM4h_samp_220x 00016249 723.LMD (36 total)
##   varLabels: id cellLine ... filename (9 total)
##   varMetadata: labelDescription
```

## 2.2 Basic data structure

### 2.2.1 Creating a FlowSet

At this stage, we use our `AnnotatedDataFrame` to create a `FlowSet` for our experiment. A `FlowSet` is simply a collection of related `FlowFrame` objects which store individual measurements. Each row in `pData` contains the file path of each individual measurement that comprises a `FlowFrame` within the `FlowSet`. As we will see later, a `FlowFrame` object can be plotted or manipulated on its own, or a `FlowSet` object can be used to plot or manipulate a collection of `FlowFrames` in a consistent fashion. In other words, in **flowCore** parlance a `FlowFrame` is a flow cytometry measurement (which corresponds to a single row in the `pData`) whereas a `FlowSet` could be an entire experiment made up of many samples (which corresponds to the collection of all rows in the `pData`). See the **flowCore** package for more in-depth explanations.

```
# Load all samples into memory as a flowSet object
flowset <- flowCore::read.flowSet(
  path=getwd(),
  dataset=1,
  transformation=FALSE,
  alter.names=TRUE,
  phenoData=phenoData
)

flowset
```

```
## A flowSet with 36 experiments.
##
## column names(15): FS.Lin SS.Lin ... FL5.Lin TIME
```

There are other ways to access flow data in R, however I have found this method to be the most elegant. Loading `FlowSet` data using an `AnnotatedDataFrame` has the following advantages:

- **Clearer experimental design:** the experiment metadata explicitly defined in the `AnnotatedDataFrame` should clarify the experimental design.
- **Better code separation:** the file paths are all located in an external tab-delimited `.txt` file rather than being defined at any given location within the source code itself. File handling can otherwise take up a significant number of lines of code.
- **Cleaner code:** with this method, an entire experiment is set up within a `FlowSet` in one command, without the need for repetitive code for file handling or cumbersome operations on individual `FlowFrame` objects.

### 2.2.2 Constructing a minimal GatingSet

While the `FlowSet` contains the actual data for the experiment, we need a means to construct and manipulate a gating tree. This is where the `GatingSet` object provided by the **flowWorkspace** package comes in. A `GatingSet` object contains a reference to a `FlowSet` experiment, as well as additional objects such as transformations (eg.: linear, log, biexponential, logicle, etc), compensation matrices, gates, as well as population statistics in the gating tree. The **flowWorkspace** tools allow for non-destructive manipulation of flow cytometry data and integrate very well with **flowJo** (which is of little help to me, but is still noteworthy).

The ideal workflow as of writing this appears to be creating an XML workspace with the **flowJo** commercial software in order to import directly into a fully constructed `GatingSet` object. Unfortunately, **flowWorkspace** does not yet appear to support importing of standard GatingML-compliant files, although as we will see, some limited support for this is offered by **flowUtils**.

**Technical documentation:**

- https://www.bioconductor.org/packages/release/bioc/html/flowWorkspace.html

We start by constructing a minimal `GatingSet` for the experiment.

```
gatingset <- flowWorkspace::GatingSet(flowset)
gatingset
```

```
## A GatingSet with 36 samples
```

This `GatingSet` is currently quite useless, but we will be making good use of it shortly.

### 2.2.3 Accessing instrument settings

While this section may at first glance appear unimportant or to be full of gibberish and utter nonsense, there are at least two good reasons for diving into the instrument settings included in `.FCS` or `.LMD` files:

- **Low-level data access: TODO: Include explanation of all the things you can find in there**
- **MIFlowCyt:** https://www.ncbi.nlm.nih.gov/pmc/articles/PMC2773297/

As per MIFlowCyt, the following list of instrument and acquisition parameters is included for informational purposes.

```
# Extract experiment's instrument metadata
keywords <- flowCore::keyword(gatingset[[1]])

# Format
keywordsDataFrame <- data.frame(BiocGenerics::t(data.frame(keywords)))
keywordsDataFrame$key <- BiocGenerics::rownames(keywordsDataFrame)
keywordsDataFrame <- keywordsDataFrame[c(2, 1)] # Reorder columns

BiocGenerics::colnames(keywordsDataFrame) <- c("key", "value")
BiocGenerics::rownames(keywordsDataFrame) <- c()

# Print
printDataFrame(
  keywordsDataFrame,
  "Instrument settings",
  fontsize=6
)
```

Table 2: Instrument settings

| key | value |
|---|---|
| FCSversion | 3 |
| X.ABSCALFACTOR | NOT SET |
| X.Acquisition.Protocol.Offset | 0000565308 |
| X.AUX_SIGNAL | N/A |
| X.BARCODE | NoRead |
| X.BASELINEOFFSET | OFF |
| X.BUILDNUMBER | 3707 |
| X.CAROUSEL | 1 |
| X.COMPENSATIONMODE | Advanced |
| X.CYTOMETERID | AG10041 |
| X.DISCRIMINATOR | FS,1 |
| X.FILEGUID | 9DF5EB9B1AA39E49A66D67F4D2E0DDC7 |
| X.HENELASER | ENABLED |
| X.LOCATION | |
| X.P10ADDRESS | 14 |
| X.P10C | ARITHMETIC |
| X.P10GAIN | 1.000000 |
| X.P10Q | FL1 Lin |
| X.P10U | |
| X.P10X | 0.0, 0.0 |
| X.P10Z | ON |
| X.P11ADDRESS | 18 |
| X.P11C | ARITHMETIC |
| X.P11GAIN | 1.000000 |
| X.P11Q | FL2 Lin |

Table 2: Instrument settings *(continued)*

| key | value |
|---|---|
| X.P11U | |
| X.P11X | 0.0, 0.0 |
| X.P11Z | ON |
| X.P12ADDRESS | 22 |
| X.P12C | ARITHMETIC |
| X.P12GAIN | 1.000000 |
| X.P12Q | FL3 Lin |
| X.P12U | |
| X.P12X | 0.0, 0.0 |
| X.P12Z | ON |
| X.P13ADDRESS | 26 |
| X.P13C | ARITHMETIC |
| X.P13GAIN | 1.000000 |
| X.P13Q | FL4 Lin |
| X.P13U | |
| X.P13X | 0.0, 0.0 |
| X.P13Z | ON |
| X.P14ADDRESS | 32 |
| X.P14C | ARITHMETIC |
| X.P14GAIN | 1.000000 |
| X.P14Q | FL5 Lin |
| X.P14U | |
| X.P14X | 0.0, 0.0 |
| X.P14Z | ON |
| X.P15ADDRESS | 4 |
| X.P15C | ARITHMETIC |
| X.P15Q | TIME |
| X.P15U | |
| X.P15X | 1023.0,500.0 |
| X.P15Z | ON |
| X.P1ADDRESS | 6 |
| X.P1C | ARITHMETIC |
| X.P1GAIN | 2.000000 |
| X.P1Q | FS Lin |
| X.P1U | |
| X.P1X | 0.0, 0.0 |
| X.P1Z | ON |
| X.P2ADDRESS | 8 |
| X.P2C | ARITHMETIC |
| X.P2GAIN | 5.000000 |
| X.P2Q | SS Lin |
| X.P2U | |
| X.P2X | 0.0, 0.0 |
| X.P2Z | ON |
| X.P3ADDRESS | 15 |
| X.P3C | GEOMETRIC |
| X.P3GAIN | 1.000000 |
| X.P3Q | FL1 Log |
| X.P3U | LOG Channels |
| X.P3Z | ON |
| X.P4ADDRESS | 19 |
| X.P4C | GEOMETRIC |
| X.P4GAIN | 1.000000 |
| X.P4Q | FL2 Log |
| X.P4U | LOGChannels |
| X.P4Z | ON |
| X.P5ADDRESS | 23 |
| X.P5C | GEOMETRIC |
| X.P5GAIN | 1.000000 |
| X.P5Q | FL3 Log |
| X.P5U | LOG Channels |
| X.P5Z | ON |
| X.P6ADDRESS | 27 |
| X.P6C | GEOMETRIC |
| X.P6GAIN | 1.000000 |
| X.P6Q | FL4 Log |
| X.P6U | |
| X.P6Z | ON |

Table 2: Instrument settings *(continued)*

| key | value |
|---|---|
| X.P7ADDRESS | 33 |
| X.P7C | GEOMETRIC |
| X.P7GAIN | 1.000000 |
| X.P7Q | FL5 Log |
| X.P7U | |
| X.P7Z | ON |
| X.P8ADDRESS | 7 |
| X.P8C | GEOMETRIC |
| X.P8GAIN | 2.000000 |
| X.P8Q | FS Log |
| X.P8U | |
| X.P8Z | ON |
| X.P9ADDRESS | 9 |
| X.P9C | GEOMETRIC |
| X.P9GAIN | 5.000000 |
| X.P9Q | SS Log |
| X.P9U | |
| X.P9Z | ON |
| X.PANEL | MitoSafe with CD44 (FL1 FITC) |
| X.RATIO_DENOMINATOR | N/A |
| X.RATIO_NUMERATOR | N/A |
| X.RATIODENOMINATORMUX | 8 |
| X.RATIONUMERATORMUX | 6 |
| X.RESAVEDFILE | RUNTIME PROTOCOL |
| X.SAMPLEID1 | 786n1_Ca1uM4h_samp_20x |
| X.SAMPLEID2 | |
| X.SAMPLEID3 | |
| X.SAMPLEID4 | |
| X.SETTINGSFILE | MPs - MTDRFM et CD44 (FITC).PRO |
| X.SETTINGSFILEDATETIME | 16-May-2018, 15:24:45 |
| X.STOPREASON | PROTOCOL |
| X.TARGETLASERPOWER | 20 mW |
| X.TUBENO | 1 |
| X.Y2KDATE | 20180516 |
| X.BEGINANALYSIS | 0 |
| X.BEGINDATA | 4325 |
| X.BEGINSTEXT | 0 |
| X.BTIM | 16:53:04 |
| X.BYTEORD | 4,3,2,1 |
| X.CELLS | |
| X.CYT | Cytomics FC 500 |
| X.DATATYPE | F |
| X.DATE | 16-May-18 |
| X.DFC1TO2 | 3.00 |
| X.DFC1TO3 | 5.20 |
| X.DFC1TO4 | 0.00 |
| X.DFC1TO5 | 0.00 |
| X.DFC2TO1 | 0.00 |
| X.DFC2TO3 | 0.00 |
| X.DFC2TO4 | 0.00 |
| X.DFC2TO5 | 0.00 |
| X.DFC3TO1 | 0.00 |
| X.DFC3TO2 | 0.00 |
| X.DFC3TO4 | 0.00 |
| X.DFC3TO5 | 0.00 |
| X.DFC4TO1 | 1.60 |
| X.DFC4TO2 | 0.00 |
| X.DFC4TO3 | 2.60 |
| X.DFC4TO5 | 25.00 |
| X.DFC5TO1 | 0.00 |
| X.DFC5TO2 | 0.00 |
| X.DFC5TO3 | 0.00 |
| X.DFC5TO4 | 0.00 |
| X.ENDANALYSIS | 0 |
| X.ENDDATA | 1118044 |
| X.ENDSTEXT | 0 |
| X.ETIM | 17:01:24 |
| X.EXP | |

Table 2: Instrument settings *(continued)*

| key | value |
|---|---|
| X.FIL | 786n1_Ca1uM4h_samp_20x 00016119 613.LMD |
| X.INST | Universite de Moncton |
| X.INSTADDRESS | |
| X.MODE | L |
| X.NEXTDATA | 0 |
| X.OP | El Guigui |
| X.P10B | 32 |
| X.P10E | 0,0 |
| X.P10G | 1.000000 |
| X.P10N | FL1.Lin |
| X.P10R | 1024 |
| X.P10S | FL1 Lin |
| X.P10V | 440 |
| X.P11B | 32 |
| X.P11E | 0,0 |
| X.P11G | 1.000000 |
| X.P11N | FL2.Lin |
| X.P11R | 1024 |
| X.P11S | FL2 Lin |
| X.P11V | 555 |
| X.P12B | 32 |
| X.P12E | 0,0 |
| X.P12G | 1.000000 |
| X.P12N | FL3.Lin |
| X.P12R | 1024 |
| X.P12S | FL3 Lin |
| X.P12V | 250 |
| X.P13B | 32 |
| X.P13E | 0,0 |
| X.P13G | 1.000000 |
| X.P13N | FL4.Lin |
| X.P13R | 1024 |
| X.P13S | FL4 Lin |
| X.P13V | 306 |
| X.P14B | 32 |
| X.P14E | 0,0 |
| X.P14G | 1.000000 |
| X.P14N | FL5.Lin |
| X.P14R | 1024 |
| X.P14S | FL5 Lin |
| X.P14V | 250 |
| X.P15B | 32 |
| X.P15E | 0,0 |
| X.P15N | TIME |
| X.P15R | 1024 |
| X.P15S | TIME |
| X.P15V | 0 |
| X.P1B | 32 |
| X.P1E | 0,0 |
| X.P1G | 2.000000 |
| X.P1N | FS.Lin |
| X.P1R | 1024 |
| X.P1S | FS Lin |
| X.P1V | 503 |
| X.P2B | 32 |
| X.P2E | 0,0 |
| X.P2G | 5.000000 |
| X.P2N | SS.Lin |
| X.P2R | 1024 |
| X.P2S | SS Lin |
| X.P2V | 508 |
| X.P3B | 32 |
| X.P3E | 0,0 |
| X.P3G | 1.000000 |
| X.P3N | FL1.Log |
| X.P3R | 1024 |
| X.P3S | FL1 Log |
| X.P3V | 440 |
| X.P4B | 32 |

Table 2: Instrument settings *(continued)*

| key | value |
|---|---|
| X.P4E | 0,0 |
| X.P4G | 1.000000 |
| X.P4N | FL2.Log |
| X.P4R | 1024 |
| X.P4S | FL2 Log |
| X.P4V | 555 |
| X.P5B | 32 |
| X.P5E | 0,0 |
| X.P5G | 1.000000 |
| X.P5N | FL3.Log |
| X.P5R | 1024 |
| X.P5S | FL3 Log |
| X.P5V | 250 |
| X.P6B | 32 |
| X.P6E | 0,0 |
| X.P6G | 1.000000 |
| X.P6N | FL4.Log |
| X.P6R | 1024 |
| X.P6S | FL4 Log |
| X.P6V | 306 |
| X.P7B | 32 |
| X.P7E | 0,0 |
| X.P7G | 1.000000 |
| X.P7N | FL5.Log |
| X.P7R | 1024 |
| X.P7S | FL5 Log |
| X.P7V | 250 |
| X.P8B | 32 |
| X.P8E | 0,0 |
| X.P8G | 2.000000 |
| X.P8N | FS.Log |
| X.P8R | 1024 |
| X.P8S | FS Log |
| X.P8V | 503 |
| X.P9B | 32 |
| X.P9E | 0,0 |
| X.P9G | 5.000000 |
| X.P9N | SS.Log |
| X.P9R | 1024 |
| X.P9S | SS Log |
| X.P9V | 508 |
| X.PAR | 15 |
| X.PROJ | |
| X.RUNNUMBER | 00016119 |
| X.SMNO | 00016119 |
| X.SRC | |
| X.SYS | CXP |
| X.TOT | 18562 |
| ACQTIME | 500.0 |
| FILENAME | /private/var/folders/23/z5mf0z_x2sj5xw7m95mdhgyh0000gn/T/RtmprUtTBD/file1797e6a9a686c/data_786 n1_786n1_Ca1uM4h_samp_20x 00016119 613.LMD |
| GUID | data_786 n1_786n1_Ca1uM4h_samp_20x 00016119 613.LMD |
| GUID.original | data/786 n1/786n1_Ca1uM4h_samp_20x 00016119 613.LMD |
| ORIGINALGUID | data_786 n1_786n1_Ca1uM4h_samp_20x 00016119 613.LMD |
| TESTFILE | MPs - MTDRFM et CD44 (FITC) |
| TESTNAME | MPs - MTDRFM et CD44 (FITC) |
| X.CYTOLIB_VERSION | 2.2.0 |
| transformation | applied |
| flowCore_.P1Rmax | 1024 |
| flowCore_.P1Rmin | 0 |
| flowCore_.P2Rmax | 1024 |
| flowCore_.P2Rmin | 0 |
| flowCore_.P3Rmax | 1024 |
| flowCore_.P3Rmin | 0 |
| flowCore_.P4Rmax | 1024 |
| flowCore_.P4Rmin | 0 |
| flowCore_.P5Rmax | 1024 |
| flowCore_.P5Rmin | 0 |
| flowCore_.P6Rmax | 1024 |

Table 2: Instrument settings *(continued)*

| key | value |
|---|---|
| flowCore_.P6Rmin | 0 |
| flowCore_.P7Rmax | 1024 |
| flowCore_.P7Rmin | 0 |
| flowCore_.P8Rmax | 1024 |
| flowCore_.P8Rmin | 0 |
| flowCore_.P9Rmax | 1024 |
| flowCore_.P9Rmin | 0 |
| flowCore_.P10Rmax | 1024 |
| flowCore_.P10Rmin | 0 |
| flowCore_.P11Rmax | 1024 |
| flowCore_.P11Rmin | 0 |
| flowCore_.P12Rmax | 1024 |
| flowCore_.P12Rmin | 0 |
| flowCore_.P13Rmax | 1024 |
| flowCore_.P13Rmin | 0 |
| flowCore_.P14Rmax | 1024 |
| flowCore_.P14Rmin | 0 |
| flowCore_.P15Rmax | 1024 |
| flowCore_.P15Rmin | 0 |

## 2.3 GatingML

```
flowEnv <- new.env()
flowUtils::read.gatingML("GatingML.xml", flowEnv)
```

### 2.3.1 Applying the gating tree onto the GatingSet

In theory, with an appropriate GatingML-compliant file this should be incredibly straightforward. However, support for standard GatingML-compliant files in **flowWorkspace** is nonexistent. From what I have been able to figure out, support for this in **flowUtils** is either broken, incomplete or improperly documented. As a result, the following code is not as elegant a demonstration as it could be, but this should not detract from the fact that the R/bioconductor open sourcee flow cytometry tools are extremely powerful.

**References:**

- https://onlinelibrary.wiley.com/doi/epdf/10.1002/cyto.a.20637

**Technical documentation:**

- http://flowcyt.sourceforge.net/gating/latest.pdf
- http://bioconductor.riken.jp/packages/3.7/bioc/manuals/flowUtils/man/flowUtils.pdf

Let's begin this mess by defining two convenience functions (which I hope to get rid of in the near future). We can then construct a gating tree from the gates defined in the GatingML file.

```
getGate <- function(gateName) {
  flowEnv[[gateName]]@filters[[1]]
}

getGateParent <- function(gateName) {
  flowEnv[[gateName]]@filters[[2]]@name
}

flowWorkspace::gs_pop_add(
  gs=gatingset,
  gate=getGate("boundary"),
  parent=getGateParent("boundary")
)

flowWorkspace::gs_pop_add(
  gs=gatingset,
  gate=getGate("nonNoise"),
  parent=getGateParent("nonNoise")
)

flowWorkspace::gs_pop_add(
  gs=gatingset,
  gate=getGate("beadsA"),
  parent=getGateParent("beadsA")
)

flowWorkspace::gs_pop_add(
  gs=gatingset,
  gate=getGate("beadsB"),
  parent=getGateParent("beadsB")
)
```

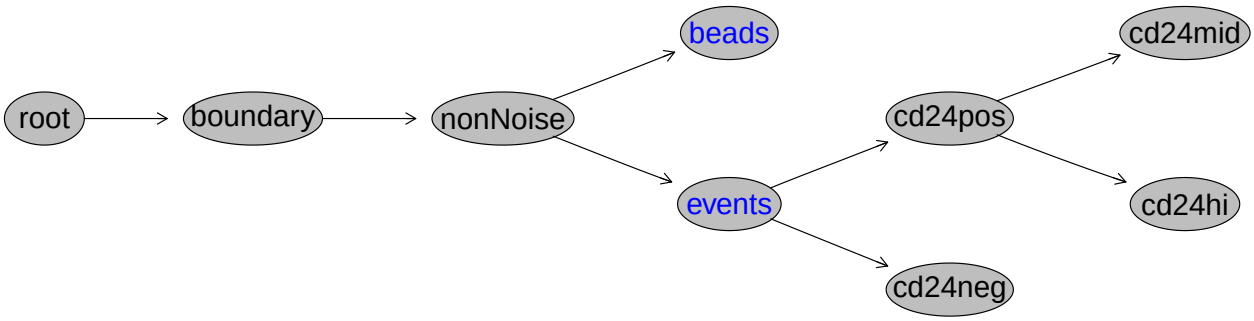
```
flowWorkspace::gs_pop_add(
  gs=gatingset,
  gate=flowWorkspace::booleanFilter(`beadsA & beadsB`),
  name="beads",
  parent="nonNoise"
)

flowWorkspace::gs_pop_add(
  gs=gatingset,
  gate=flowWorkspace::booleanFilter(`nonNoise & !beads`),
  name="events",
  parent="nonNoise"
)

flowWorkspace::gs_pop_add(
  gs=gatingset,
  gate=getGate("cd24pos"),
  parent=getGateParent("cd24pos")
)

flowWorkspace::gs_pop_add(
  gs=gatingset,
  gate=getGate("cd24neg"),
  parent=getGateParent("cd24neg")
)

flowWorkspace::gs_pop_add(
  gs=gatingset,
  gate=getGate("cd24mid"),
  parent=getGateParent("cd24mid")
)

flowWorkspace::gs_pop_add(
  gs=gatingset,
  gate=getGate("cd24hi"),
  parent=getGateParent("cd24hi")
)
```

Because individually they are completely irrelevant to our analysis, we can hide the "helper" bead count nodes from our gating tree, instead focusing on the semantically more useful `beads` boolean gate. We can then apply the gating tree to the `GatingSet`: this will also compute population statistics down the tree.

```
invisible(flowWorkspace::gs_pop_set_visibility(gatingset, "beadsA", FALSE))
invisible(flowWorkspace::gs_pop_set_visibility(gatingset, "beadsB", FALSE))

# Apply the gating tree to the underlying data
flowWorkspace::recompute(gatingset)
```

We can then plot the gating tree in order to visualize the general gating strategy for this experiment.

```
plot(gatingset, bool=TRUE)
```




## 2.4 Data normalization

There is some purely instrumental variability: beads sometimes appear at slightly different locations, so we normalize for this here.

**References:**

- https://www.ncbi.nlm.nih.gov/pmc/articles/PMC3648208/

**Technical documentation:**

- https://bioconductor.org/packages/release/bioc/manuals/flowStats/man/flowStats.pdf

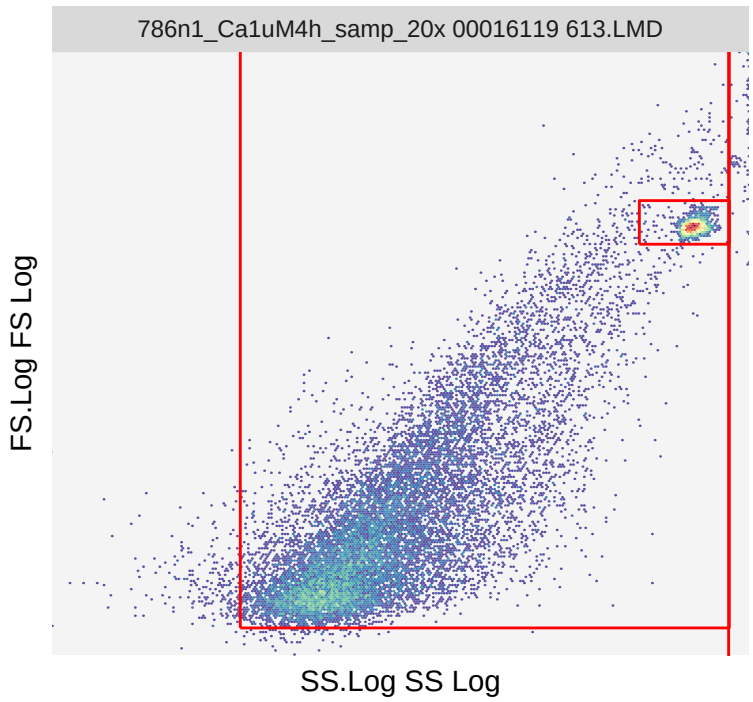
```
## Estimating landmarks for channel FS.Log ...Estimating landmarks for channel SS.Log ...
## Registering curves for parameter FS.Log ...
## Registering curves for parameter SS.Log ...
## Estimating landmarks for channel FS.Log ...Estimating landmarks for channel SS.Log ...
## Registering curves for parameter FS.Log ...
## Registering curves for parameter SS.Log ...
```

# 3 Data visualization

## 3.1 Single sample morphology plot

Gated cytometry data can be plotted with ggCyto.

```
ggcyto::ggcyto(
  data=gatingset[[1]],
  mapping=ggplot2::aes(
    x=SS.Log,
    y=FS.Log),
  subset="root"
) +
  ggplot2::geom_hex(bins=256) +
  ggplot2::scale_x_continuous(expand=c(0, 0)) +
  ggplot2::scale_y_continuous(expand=c(0, 0)) +
  ggcyto::geom_gate("boundary") +
  ggcyto::geom_gate("nonNoise") +
  ggcyto::geom_gate("beadsA") +
  ggplot2::theme(
    legend.position="none",
    panel.background=ggplot2::element_rect(fill="#F4F4F4"),
    panel.grid.major.x=ggplot2::element_blank(),
    panel.grid.minor.x=ggplot2::element_blank(),
    panel.grid.major.y=ggplot2::element_blank(),
    panel.grid.minor.y=ggplot2::element_blank(),
    panel.spacing=grid::unit(0.1, units="lines"),
    axis.text.x=ggplot2::element_blank(),
    axis.text.y=ggplot2::element_blank(),
    axis.ticks=ggplot2::element_blank())
```



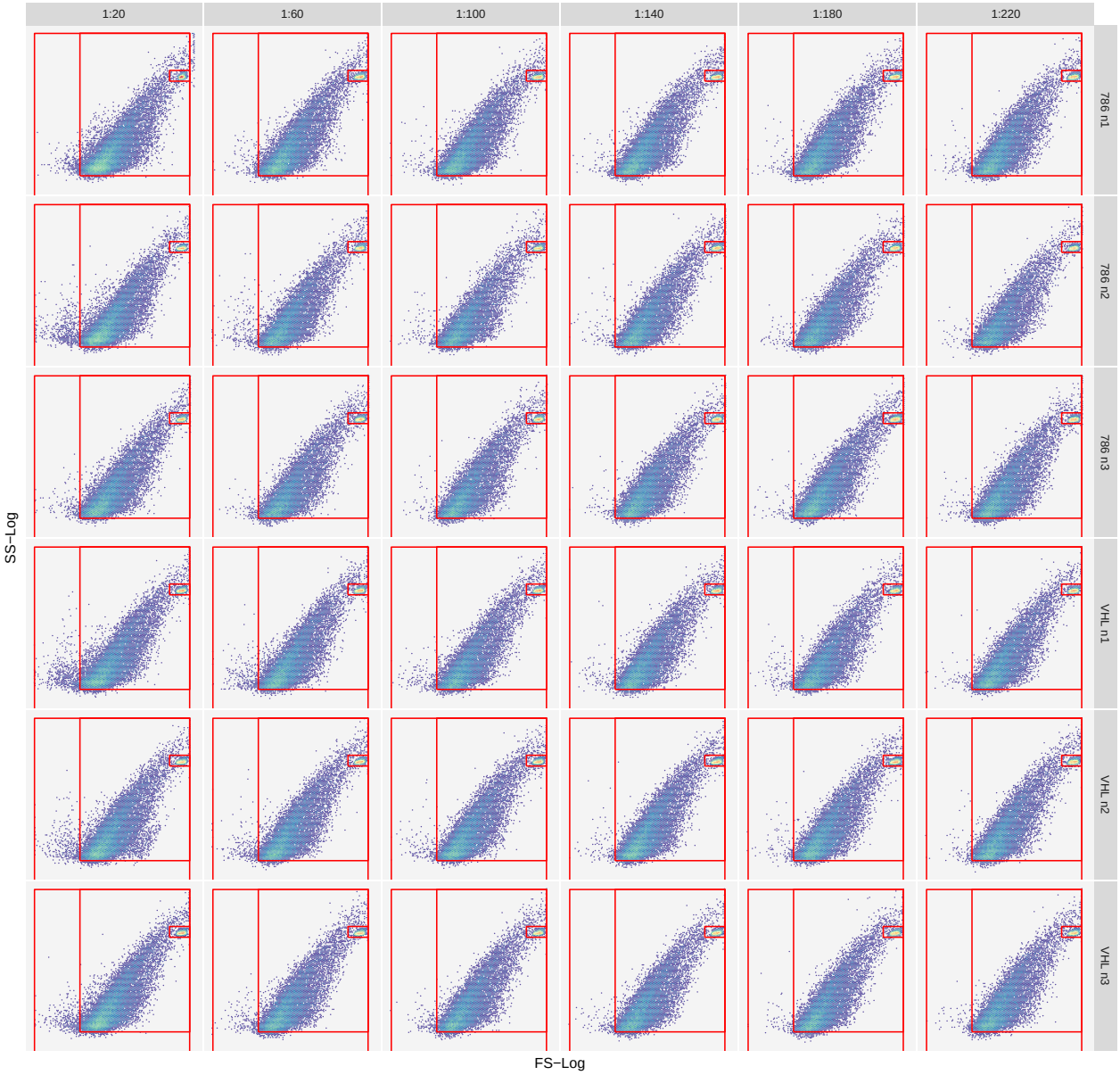
## 3.2 Whole experiment morphology plot

Let's plot the entire Gatingset so we can visualize the morphology plane on all samples simultaneously!

```
ggcyto::ggcyto(
  data=gatingset,
  mapping=ggplot2::aes(
    x=SS.Log,
    y=FS.Log),
  subset="root"
) +
  ggplot2::labs(
    title="Stratégie de gating sur les paramètres de morphologie",
    subtitle=paste0(
      "Élimination des évènements en bordure, élimination du bruit et",
      "identification des billes de comptage"),
    x="FS-Log",
    y="SS-Log") +
  ggplot2::geom_hex(bins=128) +
  ggcyto::geom_gate("boundary") +
  ggcyto::geom_gate("nonNoise") +
  ggcyto::geom_gate("beadsA") +
  ggplot2::facet_grid(
    rows=dplyr::vars(facetRow),
    cols=dplyr::vars(facetCol),
    labeller=ggplot2::as_labeller(facetLabels)) +
  ggplot2::theme(
    legend.position="none",
    panel.background=ggplot2::element_rect(fill="#F4F4F4"),
    panel.grid.major.x=ggplot2::element_blank(),
    panel.grid.minor.x=ggplot2::element_blank(),
    panel.grid.major.y=ggplot2::element_blank(),
    panel.grid.minor.y=ggplot2::element_blank(),
    panel.spacing=grid::unit(0.1, units="lines"),
    axis.text.x=ggplot2::element_blank(),
    axis.text.y=ggplot2::element_blank(),
    axis.ticks=ggplot2::element_blank())
```

**Stratégie de gating sur les paramètres de morphologie**

Élimination des évènements en bordure, élimination du bruit etidentification des billes de comptage



### 3.3 Auxiliary bead gate visualization

Visualizing the auxiliary bead gates:

```
gatingset <- flowStats::normalize(
  data=gatingset,
  populations=c("beadsB"),
  dims=c("SS.Lin"),
  minCountThreshold=100
)
```

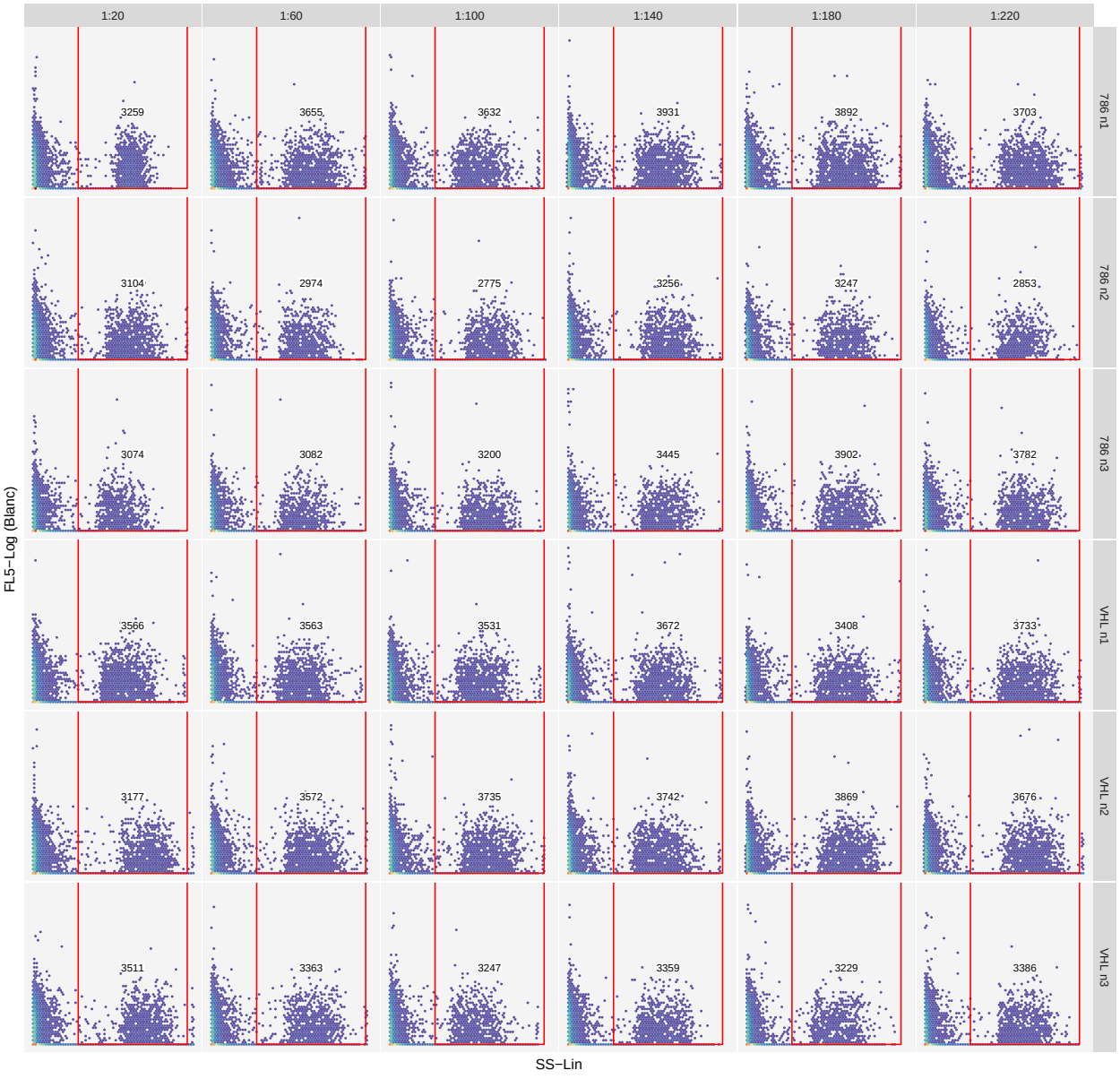
```
## cloning the gatingSet...
## Normalize beadsB
## Estimating landmarks for channel SS.Lin ...
## Registering curves for parameter SS.Lin ...
## normalizing sample 1
## normalizing sample 2
## normalizing sample 3
## normalizing sample 4
## normalizing sample 5
## normalizing sample 6
## normalizing sample 7
## normalizing sample 8
## normalizing sample 9
## normalizing sample 11
## normalizing sample 12
## normalizing sample 13
## normalizing sample 14
## normalizing sample 15
## normalizing sample 16
## normalizing sample 17
## normalizing sample 18
## normalizing sample 19
## normalizing sample 20
## normalizing sample 21
## normalizing sample 22
## normalizing sample 23
## normalizing sample 24
## normalizing sample 25
## normalizing sample 26
## normalizing sample 27
```

```
## normalizing sample 28
## normalizing sample 29
## normalizing sample 30
## normalizing sample 31
## normalizing sample 32
## normalizing sample 33
## normalizing sample 34
## normalizing sample 35
## normalizing sample 36
## normalizing sample 37
## done!
```

```
ggcyto::ggcyto(
  data=gatingset,
  mapping=ggplot2::aes(
    x=SS.Lin,
    y=FL5.Log),
  subset="nonNoise"
) +
  ggplot2::labs(
    title="Stratégie de gating pour l'élimination des beads",
    subtitle="Gating auxiliaire des billes de comptage",
    x="SS-Lin",
    y="FL5-Log (Blanc)") +
  ggplot2::geom_hex(bins=64) +
  ggcyto::geom_gate("beadsB") +
  ggcyto::geom_stats(
    size=2.7,
    type="count") +
  ggplot2::facet_grid(
    rows=dplyr::vars(facetRow),
    cols=dplyr::vars(facetCol),
    labeller=ggplot2::as_labeller(facetLabels)) +
  ggplot2::theme(
    legend.position="none",
    panel.background=ggplot2::element_rect(fill="#F4F4F4"),
    panel.grid.major.x=ggplot2::element_blank(),
    panel.grid.minor.x=ggplot2::element_blank(),
    panel.grid.major.y=ggplot2::element_blank(),
    panel.grid.minor.y=ggplot2::element_blank(),
    panel.spacing=grid::unit(0.1, units="lines"),
    axis.text.x=ggplot2::element_blank(),
    axis.text.y=ggplot2::element_blank(),
    axis.ticks=ggplot2::element_blank())
```

**Stratégie de gating pour l'élimination des beads**

Gating auxiliaire des billes de comptage



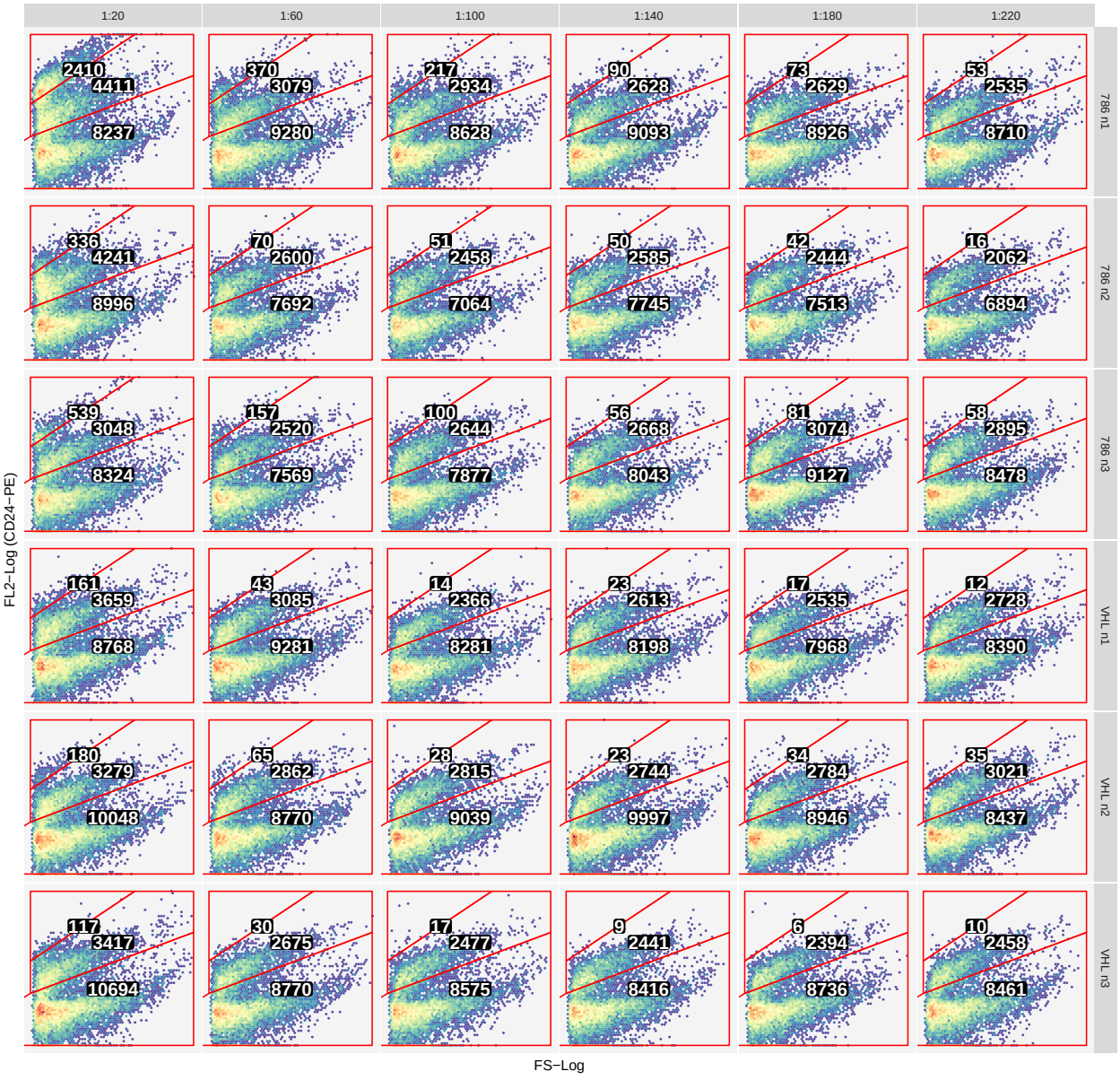
## 3.4 Whole experiment fluorescence with event counts

And finally, let's visualize the non-bead events on the fluorescence channel (anti-CD24 antibodies)

```
ggcyto::ggcyto(
  data=gatingset,
  mapping=ggplot2::aes(
    x=FS.Log,
    y=FL2.Log),
  subset="events"
) +
  ggplot2::labs(
    title="Quantification des microparticules marquées au CD24-PE",
    subtitle=paste0(
      "Comparaison des 786-O (n=3) et des 786-O/VHL (n=3). ",
      "[Gated: events]"
    ),
    x="FS-Log",
    y="FL2-Log (CD24-PE)") +
  ggplot2::geom_hex(bins=64) +
  ggcyto::geom_gate(c(
    "cd24hi",
    "cd24mid",
    "cd24neg")) +
  ggcyto::geom_stats(
    color="#FFFFFF",
    fill="#000000",
    size=5,
    fontface="bold",
    type="count") +
  ggplot2::facet_grid(
    rows=dplyr::vars(facetRow),
    cols=dplyr::vars(facetCol),
    labeller=ggplot2::as_labeller(facetLabels)) +
  ggplot2::theme(
    legend.position="none",
    panel.background=ggplot2::element_rect(fill="#F4F4F4"),
    panel.grid.major.x=ggplot2::element_blank(),
    panel.grid.minor.x=ggplot2::element_blank(),
    panel.grid.major.y=ggplot2::element_blank(),
    panel.grid.minor.y=ggplot2::element_blank(),
    panel.spacing=grid::unit(0.1, units="lines"),
    axis.text.x=ggplot2::element_blank(),
    axis.text.y=ggplot2::element_blank(),
    axis.ticks=ggplot2::element_blank())
```

**Quantification des microparticules marquées au CD24−PE**

Comparaison des 786−O (n=3) et des 786−O/VHL (n=3). [Gated: events]



# 4 Analysis

## 4.1 Data-driven methods

### 4.1.1 Fingerprinting

Fingerprinting is a quick and easy way to observe population-agnostic, data-driven differences between a large number of samples.

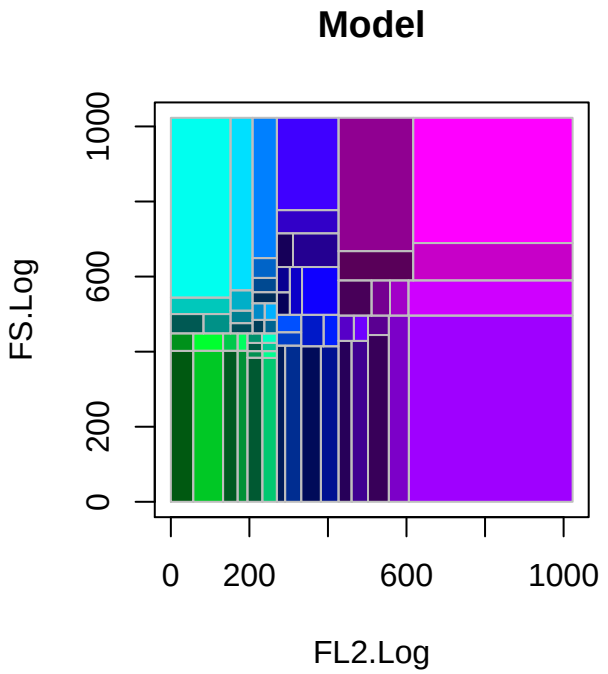
**References:**

- https://www.ncbi.nlm.nih.gov/pubmed/19956416

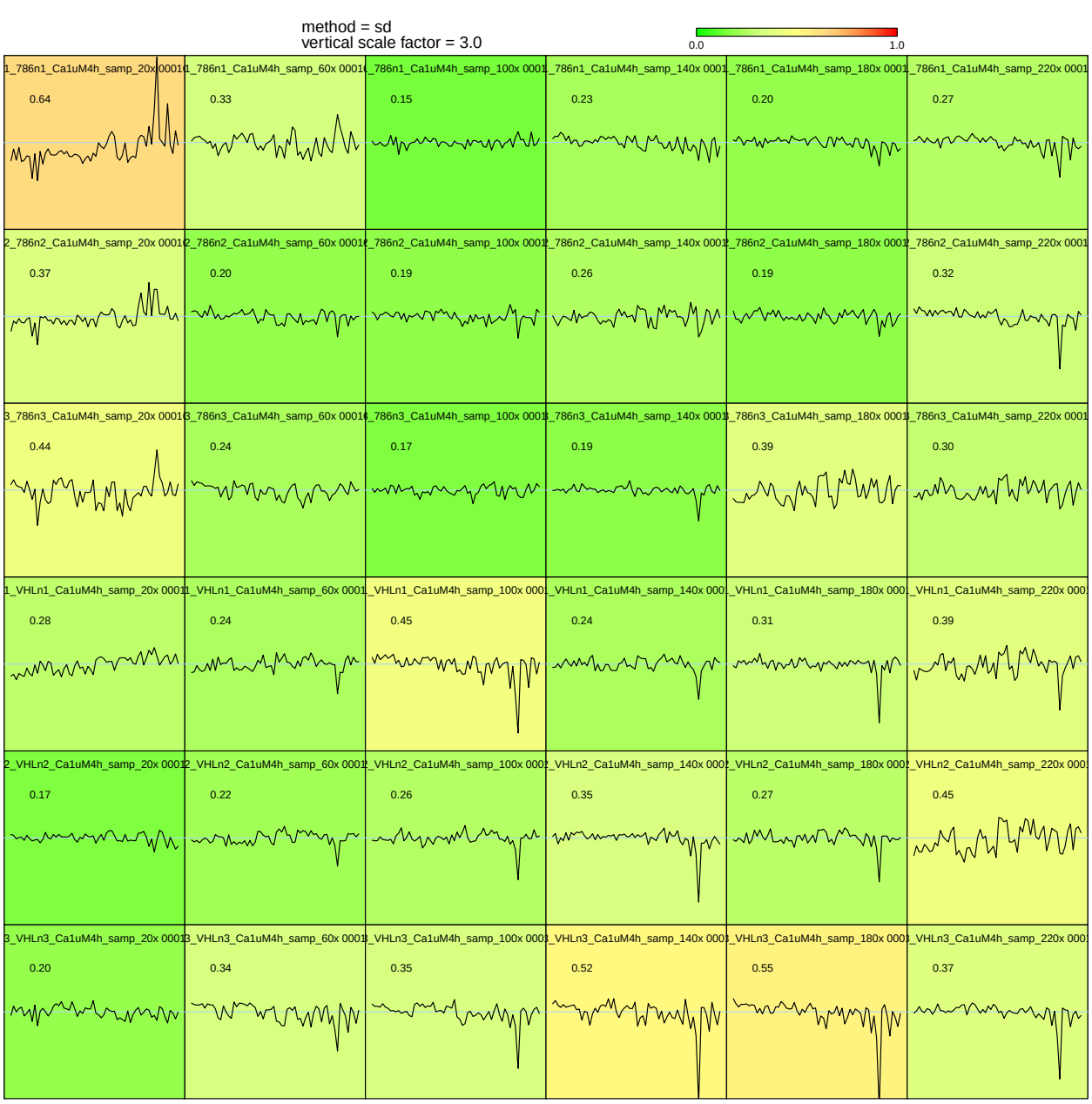
**Technical documentation:**

*https://bioconductor.org/packages/release/bioc/vignettes/flowFP/inst/doc/flowFP_HowTo.pdf

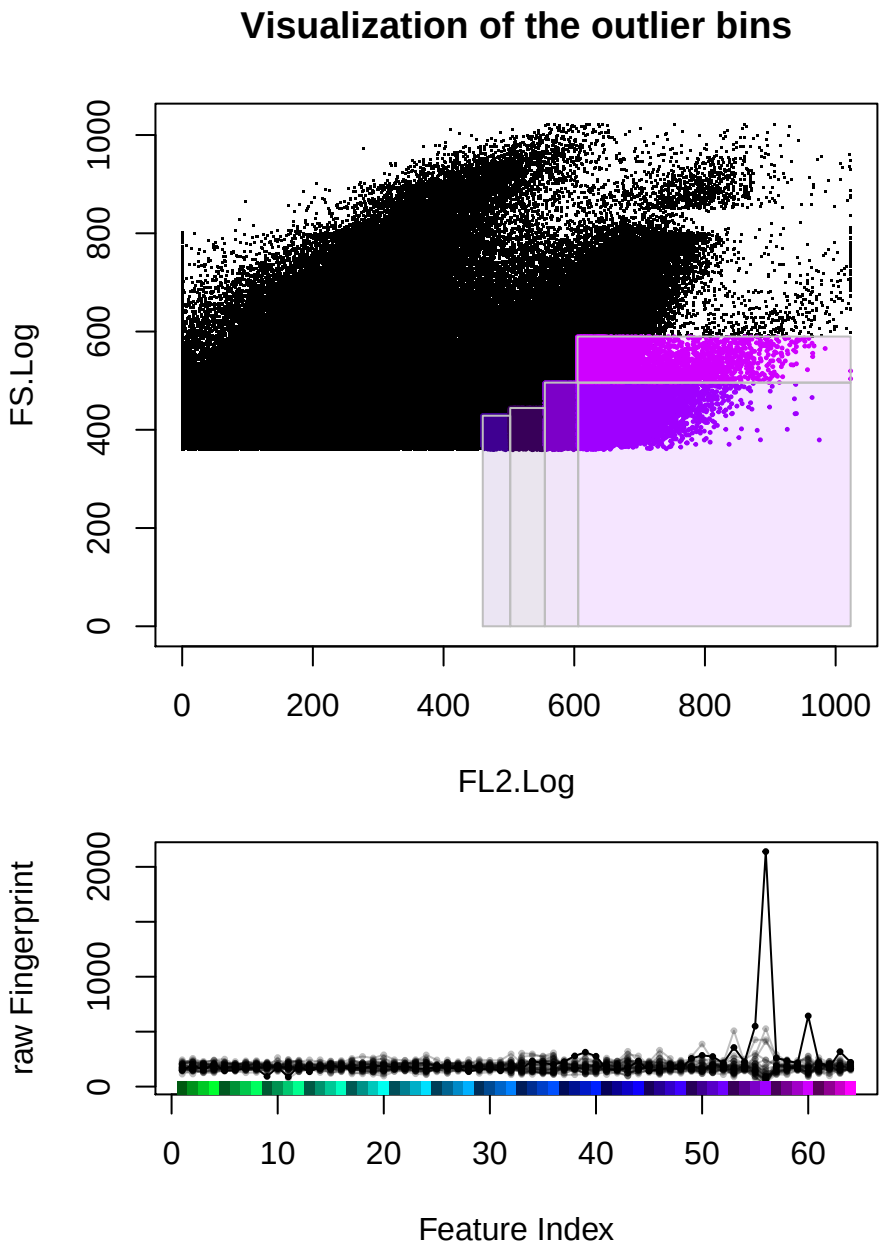
```
# Remove all irrelevant populations, such as beads and noise
fingerprintingData <- flowWorkspace::gs_pop_get_data(gatingset, "events")
fingerprintingData <- flowWorkspace::cytoset_to_flowSet(fingerprintingData)

# Generate a fingerprinting model with the data in the "events"-gated FlowSet
fingerprintingModel <- flowFP::flowFPModel(
  fcs=fingerprintingData,
  name="CD24/FS.Log Model",
  parameters=c("FS.Log", "FL2.Log"),
  nRecursions=6
)

# Generate fingerprints for all samples
fingerprints <- flowFP::flowFP(
  fcs=fingerprintingData,
  model=fingerprintingModel
)

# Visualize model
plot(fingerprintingModel)
```


Model
FS.Log
0
200
600
1000
0
200
600
1000
FL2.Log

```r
# Detect outliers
plot(fingerprints, type="qc", main="Outliers")
```



```r
# Visualize a specific sample and its outlier bins
plot(
  fingerprints,
  fingerprintingData,
  hi=1,
  showbins=c(50, 53, 55, 56, 60),
  pch=20,
  cex=0.3,
  main="Visualization of the outlier bins"
)
```



## 4.2 Swarm deconvolution

This is a novel application of a deconvolution strategy that has been validated in entirely different contexts.

**Reference:**

- http://tinyheero.github.io/2016/01/03/gmm-em.html

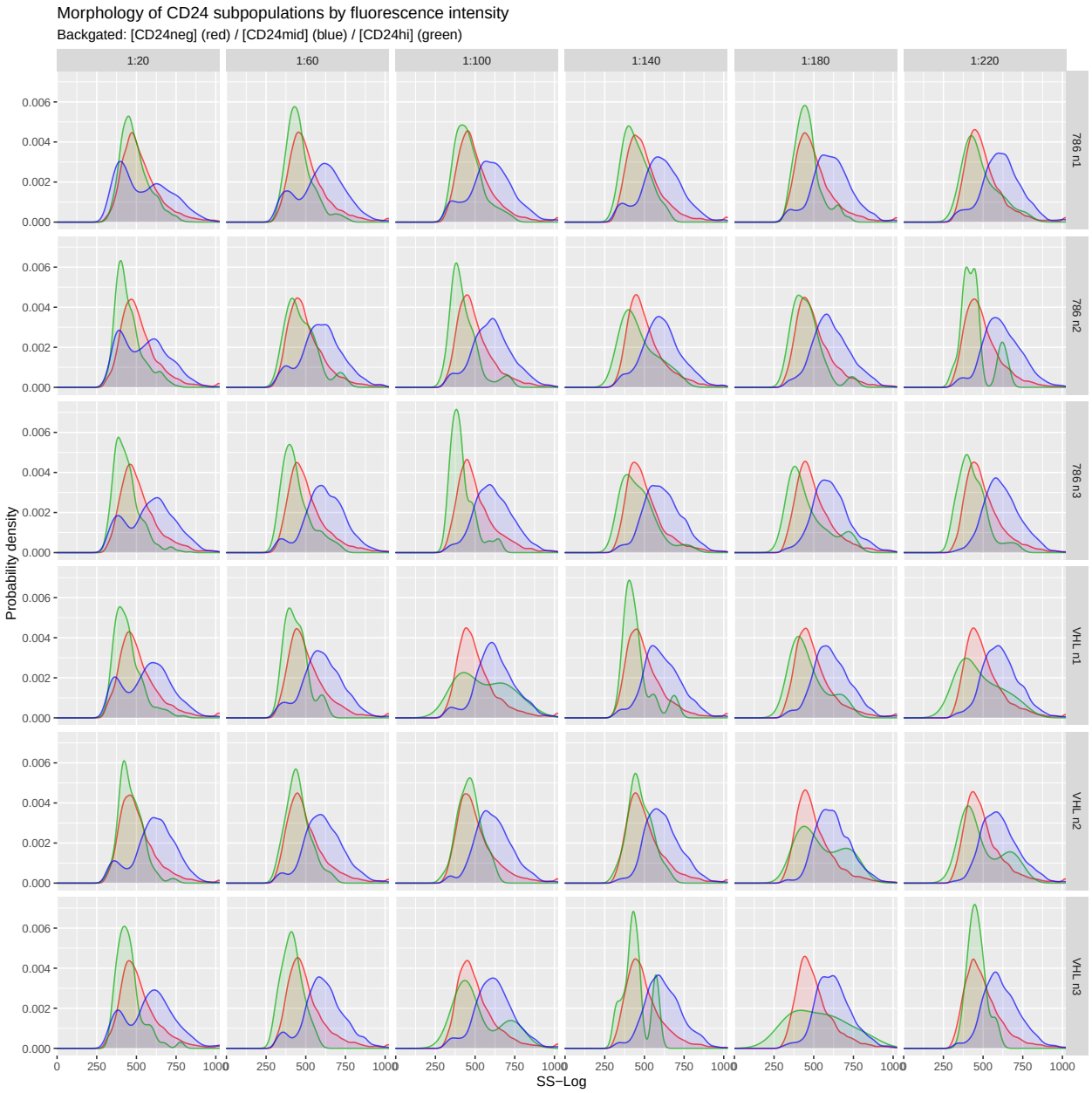
DECONVOLUTION TOOLS:

- http://journals.plos.org/plosone/article?id=10.1371/journal.pone.0067620
- https://bioconductor.org/packages/release/bioc/vignettes/flowFit/inst/doc/HowTo-flowFit.pdf

### 4.2.1 Visualizing the swarm effect

We can back-gate any of our fluorescent populations into a morphology axis:

```
# Morphology characteristics of fluorescence-gated data can be visualized to
# gain quantitative insight into the magnitude of per-subpopulation swarm effect
ggplot2::ggplot(
  data=flowWorkspace::gs_pop_get_data(gatingset, "events"),
  mapping=ggplot2::aes(x=SS.Log)
) +
  ggplot2::labs(
    title="Morphology of CD24 subpopulations by fluorescence intensity",
    subtitle="Backgated: [CD24neg] (red) / [CD24mid] (blue) / [CD24hi] (green)",
    x="SS-Log",
    y="Probability density") +
  ggplot2::scale_x_continuous(
    limits=c(0, 1023),
    expand=c(0, 0)) +
  ggplot2::geom_density(
    data=flowWorkspace::gs_pop_get_data(gatingset, "cd24neg"),
    color="#FF0000AA",
    fill="red",
    alpha=0.1) +
  ggplot2::geom_density(
    data=flowWorkspace::gs_pop_get_data(gatingset, "cd24hi"),
    color="#00AA00AA",
    fill="#00AA00",
    alpha=0.1) +
  ggplot2::geom_density(
    data=flowWorkspace::gs_pop_get_data(gatingset, "cd24mid"),
    color="#0000FFAA",
    fill="blue",
    alpha=0.1) +
  ggplot2::facet_grid(
    rows=dplyr::vars(facetRow),
    cols=dplyr::vars(facetCol),
    labeller=ggplot2::as_labeller(facetLabels))
```


Morphology of CD24 subpopulations by fluorescence intensity
Backgated: [CD24neg] (red) / [CD24mid] (blue) / [CD24hi] (green)
1:20
1:60
1:100
1:140
1:180
1:220
786 n1
786 n2
786 n3
VHL n1
VHL n2
VHL n3
Probability density
0.006
0.004
0.002
0.000
0
250
500
750
1000
SS−Log

It might be easier to visualize this by combining data from all triplicates:

```
cellLineOrdering <- c("786-O", "VHL")
dilutionOrdering <- c("20", "60", "100", "140", "180", "220")

eventsData <- flowWorkspace::gs_pop_get_data(gatingset, "events")
eventsData <- flowWorkspace::cytoset_to_flowSet(eventsData)
cd24negData <- flowWorkspace::gs_pop_get_data(gatingset, "cd24neg")
cd24negData <- flowWorkspace::cytoset_to_flowSet(cd24negData)
cd24midData <- flowWorkspace::gs_pop_get_data(gatingset, "cd24mid")
cd24midData <- flowWorkspace::cytoset_to_flowSet(cd24midData)
cd24hiData <- flowWorkspace::gs_pop_get_data(gatingset, "cd24hi")
cd24hiData <- flowWorkspace::cytoset_to_flowSet(cd24hiData)

eventsData@phenoData@data$dilution <- factor(
  eventsData@phenoData@data$dilution,
  levels=dilutionOrdering)
cd24negData@phenoData@data$dilution <- factor(
  cd24negData@phenoData@data$dilution,
  levels=dilutionOrdering)
cd24midData@phenoData@data$dilution <- factor(
  cd24midData@phenoData@data$dilution,
  levels=dilutionOrdering)
cd24hiData@phenoData@data$dilution <- factor(
  cd24hiData@phenoData@data$dilution,
  levels=dilutionOrdering)

eventsData@phenoData@data$cellLine <- factor(
  eventsData@phenoData@data$cellLine,
  levels=cellLineOrdering)
cd24negData@phenoData@data$cellLine <- factor(
  cd24negData@phenoData@data$cellLine,
  levels=cellLineOrdering)
cd24midData@phenoData@data$cellLine <- factor(
  cd24midData@phenoData@data$cellLine,
  levels=cellLineOrdering)
cd24hiData@phenoData@data$cellLine <- factor(
  cd24hiData@phenoData@data$cellLine,
  levels=cellLineOrdering)

ggplot2::ggplot(
  data=eventsData,
  mapping=ggplot2::aes(x=SS.Log)
) +
  ggplot2::labs(
    title="Morphology of CD24 subpopulations by fluorescence intensity",
    subtitle="Backgated: [CD24neg] (red) / [CD24mid] (blue) / [CD24hi] (green)",
    x="SS-Log",
    y="Probability density") +
  ggplot2::scale_x_continuous(
    limits=c(0, 1023),
    expand=c(0, 0)) +
  ggplot2::geom_density(
    data=cd24negData,
```

```
    color="#FF0000AA",
    fill="red",
    alpha=0.1) +
  ggplot2::geom_density(
    data=cd24hiData,
    color="#00AA00AA",
    fill="#00AA00", # Dark green
    alpha=0.1) +
  ggplot2::geom_density(
    data=cd24midData,
    color="#0000FFAA",
    fill="blue",
    alpha=0.1) +
  ggplot2::facet_grid(
    rows=dplyr::vars(cellLine),
    cols=dplyr::vars(dilution),
    labeller=ggplot2::as_labeller(facetLabels))
```