

immunoClust - Automated Pipeline for Population Detection in Flow Cytometry

*Till Sörensen*¹

¹till-
antoni.soerensen@charite.de

August 13, 2020

Contents

1	Licensing	2
2	Overview.	2
3	Getting started	2
4	Example Illustrating the immunoClust Pipeline	3
4.1	Cell Event Clustering	3
4.2	Meta Clustering	7
4.3	Meta Annotation	8
5	Session Info	12

1 Licensing

Under the Artistic License, you are free to use and redistribute this software. However, we ask you to cite the following paper if you use this software for publication.

Sörensen, T., Baumgart, S., Durek, P., Grützkau, A. and Häupl, T.
immunoClust - an automated analysis pipeline for the identification of immunophenotypic signatures in high-dimensional cytometric datasets.
Cytometry A (accepted).

2 Overview

immunoClust presents an automated analysis pipeline for uncompensated fluorescence and mass cytometry data and consists of two parts. First, cell events of each sample are grouped into individual clusters (cell-clustering). Subsequently, a classification algorithm assorts these cell event clusters into populations comparable between different samples (meta-clustering). The clustering of cell events is designed for datasets with large event counts in high dimensions as a global unsupervised method, sensitive to identify rare cell types even when next to large populations. Both parts use model-based clustering with an iterative Expectation Maximization (EM) algorithm and the Integrated Classification Likelihood (ICL) to obtain the clusters.

The cell-clustering process fits a mixture model with t -distributions. Within the clustering process a optimisation of the *asinh*-transformation for the fluorescence parameters is included.

The meta-clustering fits a Gaussian mixture model for the meta-clusters, where adjusted Bhattacharyya-Coefficients give the probability measures between cell- and meta-clusters.

Several plotting routines are available visualising the results of the cell- and meta-clustering process. Additional helper-routines to extract population features are provided.

3 Getting started

The installation on *immunoClust* is normally done within the Bioconductor.

The core functions of *immunoClust* are implemented in C/C++ for optimal utilization of system resources and depend on the GNU Scientific Library (GSL) and Basic Linear Subprogram (BLAS). When installing *immunoClust* form source using Rtools be aware to adjust the GSL library and include pathes in `src/Makevars.in` or `src/Makevars.win` (on Windows systems) repectively to the correct installation directory of the GSL-library on the system.

immunoClust relies on the *flowFrame* structure imported from the *flowCore*-package for accessing the measured cell events from a flow cytometer device.

4 Example Illustrating the immunoClust Pipeline

The functionality of the immunoClust pipeline is demonstrated on a dataset of blood cell samples of defined composition that were depleted of particular cell subsets by magnetic cell sorting. Whole blood leukocytes taken from three healthy individuals, which were experimentally modified by the depletion of one particular cell type per sample, including granulocytes (using CD15-MACS-beads), monocytes (using CD14-MACS-beads), T lymphocytes (CD3-MACS-beads), T helper lymphocytes (using CD4-MACS-beads) and B lymphocytes (using CD19-MACS-beads).

The example datasets contain reduced (10.000 cell-events) of the first Flow Cytometry (FC) sample in `dat.fcs` and the *immunoClust* cell-clustering results of all 5 reduced FC samples for the first donor in `dat.exp`. The full sized dataset is published and available under <http://flowrepository.org/id/FR-FCM-ZZWB>.

4.1 Cell Event Clustering

```
> library(immunoClust)
```

The cell-clustering is performed by the `cell.process` function for each FC sample separately. Its major input are the measured cell-events in a *flowFrame*-object imported from the *flowCore*-package.

```
> data(dat.fcs)
> dat.fcs

flowFrame object '2d36b4cf-da0f-4b8d-9a4c-fc7e4f5fccc8'
with 10000 cells and 7 observables:
      name desc  range  minRange maxRange
$P2      FSC-A <NA> 262144   0.00000  262143
$P5      SSC-A <NA> 262144 -111.00000 262143
$P8      FITC-A CD14 262144 -111.00000 262143
$P9      PE-A  CD19 262144 -111.00000 262143
$P12     APC-A CD15 262144 -111.00000 262143
$P13     APC-Cy7-A CD4 262144 -111.00000 262143
$P14 Pacific Blue-A CD3 262144 -98.93999 262143
171 keywords are stored in the 'description' slot
```

In the `parameters` argument the parameters (named as observables in the *flowFrame*) used for cell-clustering are specified. When omitted all determined parameters are used.

```
> pars=c("FSC-A", "SSC-A", "FITC-A", "PE-A", "APC-A", "APC-Cy7-A", "Pacific Blue-A")
> res.fcs <- cell.process(dat.fcs, parameters=pars)
```

The `summary` method for an *immunoClust*-object gives an overview of the clustering results.

```
> summary(res.fcs)

** Experiment Information **
Experiment name: immunoClust Experiment
Data Filename:   fcs/12443.fcs
```

immunoClust

```
Parameters:  FSC-A SSC-A FITC-A PE-A APC-A APC-Cy7-A Pacific Blue-A
Description:  NA NA CD14 CD19 CD15 CD4 CD3

** Data Information **
Number of observations: 10000
Number of parameters: 7
Removed from above: 318 (3.18%)
Removed from below: 0 (0%)

** Transformation Information **
htrans-A: 0.000000 0.000000 0.008051 0.003421 0.006796 0.009307 0.017760
htrans-B: 0.000000 0.000000 0.000000 0.000000 0.000000 0.000000 0.000000
htrans-decade: -1

** Clustering Summary **
ICL bias: 0.30
Number of clusters: 12
Cluster      Proportion  Observations
1            0.638834      6182
2            0.014925      142
3            0.007239       69
4            0.029189      284
5            0.007357       70
6            0.049502      482
7            0.040287      391
8            0.036719      365
9            0.053497      514
10           0.005068       49
11           0.002754       22
12           0.114626     1112

Min.         0.002754       22
Max.         0.638834     6182

** Information Criteria **
Log likelihood: -250232.2 -250363 -169202.5
BIC: -250232.2
ICL: -250363
```

With the `bias` argument of the `cell.process` function the number of clusters in the final model is controlled.

```
> res2 <- cell.process(dat.fcs, bias=0.25)
> summary(res2)

** Experiment Information **
Experiment name: immunoClust Experiment
Data Filename:  fcs/12443.fcs
Parameters:  FSC-A SSC-A FITC-A PE-A APC-A APC-Cy7-A Pacific Blue-A
Description:  NA NA CD14 CD19 CD15 CD4 CD3

** Data Information **
```

immunoClust

```
Number of observations: 10000
Number of parameters: 7
Removed from above: 318 (3.18%)
Removed from below: 0 (0%)

** Transformation Information **
htrans-A: 0.000000 0.000000 0.010000 0.010000 0.010000 0.010000 0.010000
htrans-B: 0.000000 0.000000 0.000000 0.000000 0.000000 0.000000 0.000000
htrans-decade: -1

** Clustering Summary **
ICL bias: 0.25
Number of clusters: 20
Cluster      Proportion  Observations
  1          0.030037      295
  2          0.106600      936
  3          0.217235     1960
  4          0.005143       50
  5          0.007271       70
  6          0.009625       94
  7          0.002071       19
  8          0.029521      275
  9          0.001373       13
 10          0.020028      200
 11          0.023697      229
 12          0.013254      125
 13          0.311503     3249
 14          0.087765      861
 15          0.036060      352
 16          0.054047      519
 17          0.004577       46
 18          0.018641      178
 19          0.008875       82
 20          0.012678      129

      Min.      0.001373      13
      Max.      0.311503     3249

** Information Criteria **
Log likelihood: -253842 -255707.8 -173229.1
BIC: -253842
ICL: -255707.8
```

An ICL-bias of 0.3 is reasonable for fluorescence cytometry data based on our experiences, whereas the number of clusters increase dramatically when a `bias` below 0.2 is applied. A principal strategy for the ICL-bias in the whole pipeline is the use of a moderately small `bias` (0.2 - 0.3) for cell-clustering and to optimise the `bias` on meta-clustering level to retrieve the common populations across all samples.

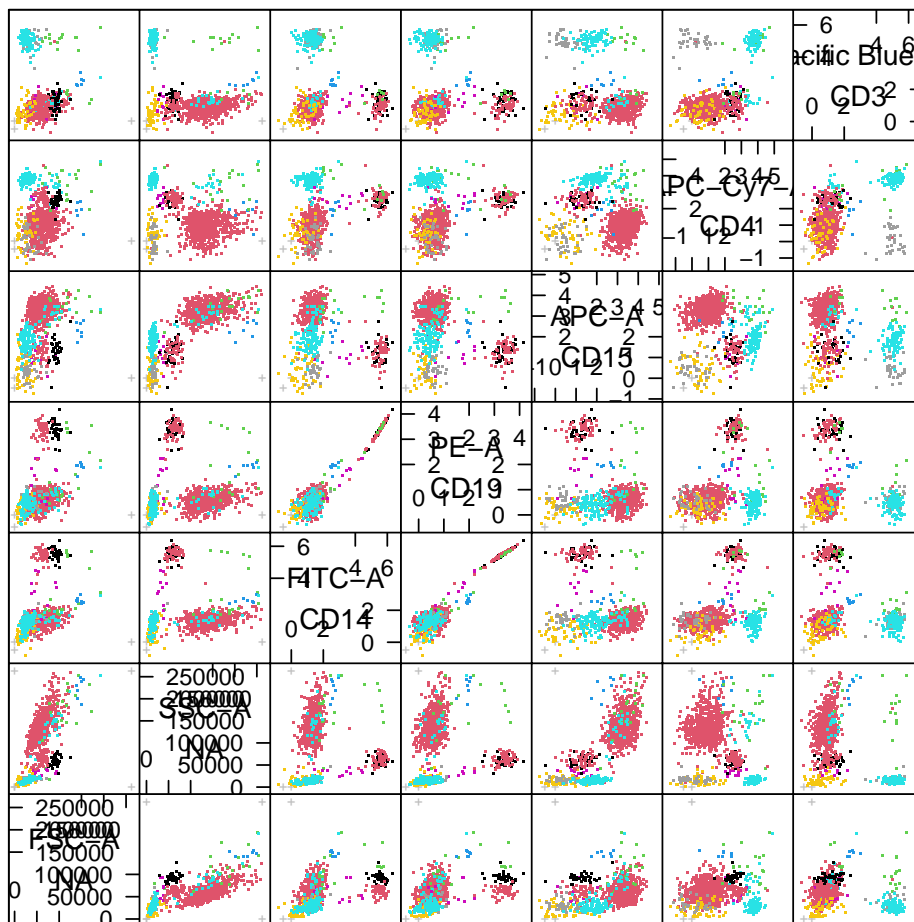
For plotting the clustering results on cell event level, the optimised `asinh`-transformation has to be applied to the raw FC data first.

immunoClust

```
> dat.transformed <- trans.ApplyToData(res.fcs, dat.fcs)
```

A scatter plot matrix of all used parameters for clustering is obtained by the `splom` method.

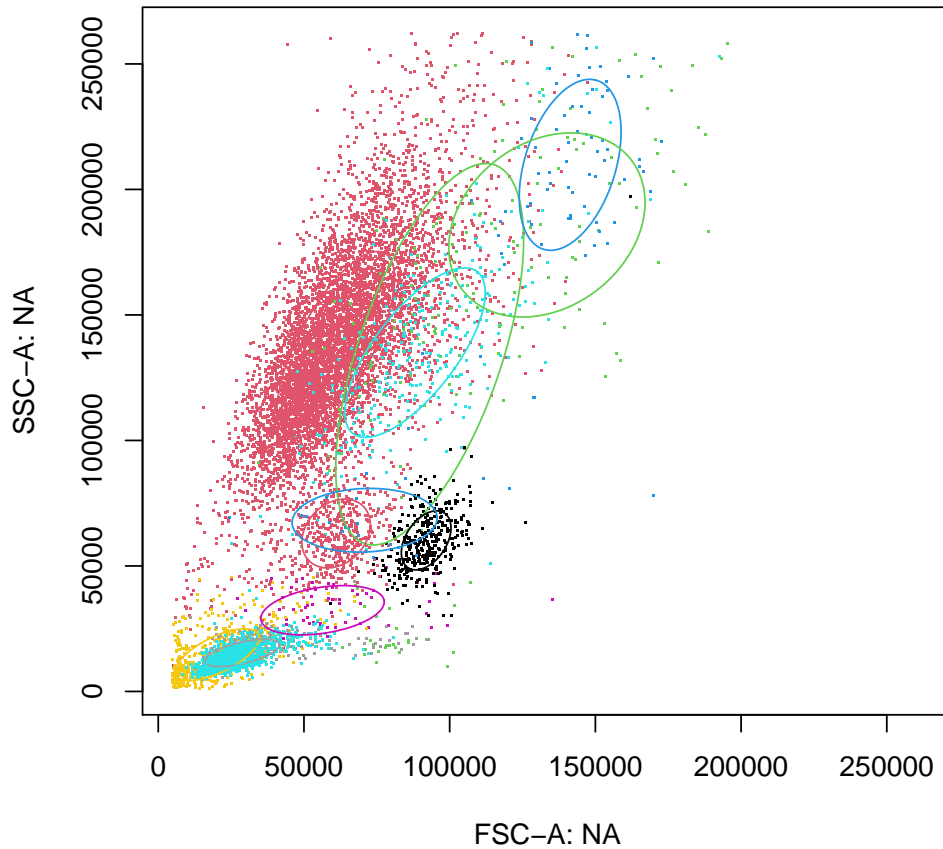
```
> splom(res.fcs, dat.transformed, N=1000)
```



Scatter Plot Matrix

For a scatter plot of 2 particular parameters the `plot` method can be used, where parameters of interest are specified in the `subset` argument.

```
> plot(res.fcs, data=dat.transformed, subset=c(1,2))
```



4.2 Meta Clustering

For meta-clustering the cell-clustering results of all FC samples obtained by the `cell.process` function are collected in a vector of `immunoClust`-objects and processed by the `meta.process` function.

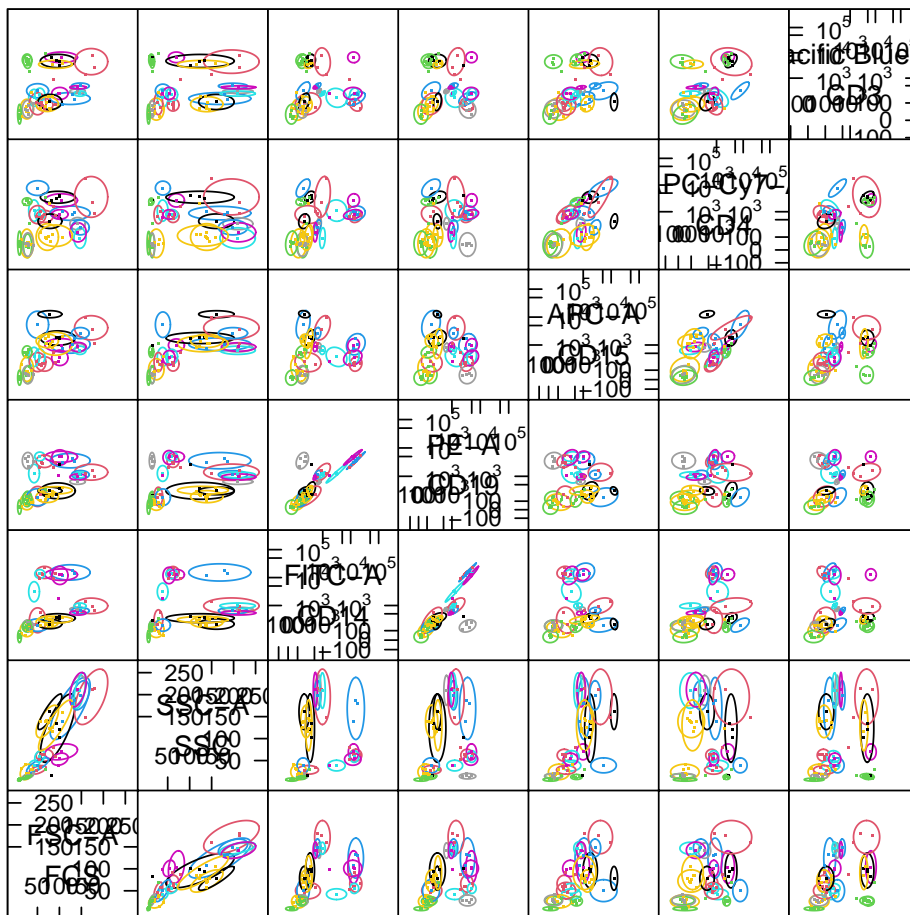
```
> data(dat.exp)
> meta<-meta.process(dat.exp, meta.bias=0.3)
```

The obtained `immunoMeta`-object contains the meta-clustering result in `$res.clusters`, and the used cell-clusters information in `$dat.clusters`. Additionally, the clusters can be structures manually in a hierarchical manner using methods of the `immunoMeta`-object.

A scatter plot matrix of the meta-clustering is obtained by the `plot` method.

```
> plot(meta, c())
```

.all



In these scatter plots each cell-cluster is marked by a point of its centre. With the default `plot.ellipse=TRUE` argument the meta-clusters are outlined by ellipses of the 90% quantile.

4.3 Meta Annotation

We take a look on the event numbers of all meta-clusters in each sample

```
> cls <- clusters(meta,c())
> events(meta,cls)

      cls-1 cls-2 cls-3 cls-4 cls-5 cls-6 cls-7 cls-8 cls-9 cls-10 cls-11
exp-1  898  389   50    0    0  344    0  143   71  1107    0
exp-2   0 1079    0  173  102  695  926    8  145  3425   220
exp-3   0  574    0    0    0  780  452  199    0  1585    0
exp-4  761  433   62    0    0  527  331    0    0    0    0
exp-5  950   46   94    0    0  400  325    0    0    0    0

      cls-12 cls-13 cls-14 cls-15 cls-16 cls-17 cls-18 cls-19 cls-20 cls-21
exp-1     0     0  6459   70     0     0   151     0     0     0
exp-2  1447   923     0     0    24   103  495    77     0     0
```


immunoClust

```
exp-3    0    0  5717    0    0    10   247    0    0   132
exp-4    0    0  7280    0    0    0   247    0   95    0
exp-5    0    0  7417    0    0    0   278    0    0    0
  cls-22
exp-1    0
exp-2    0
exp-3    40
exp-4    0
exp-5    0
```

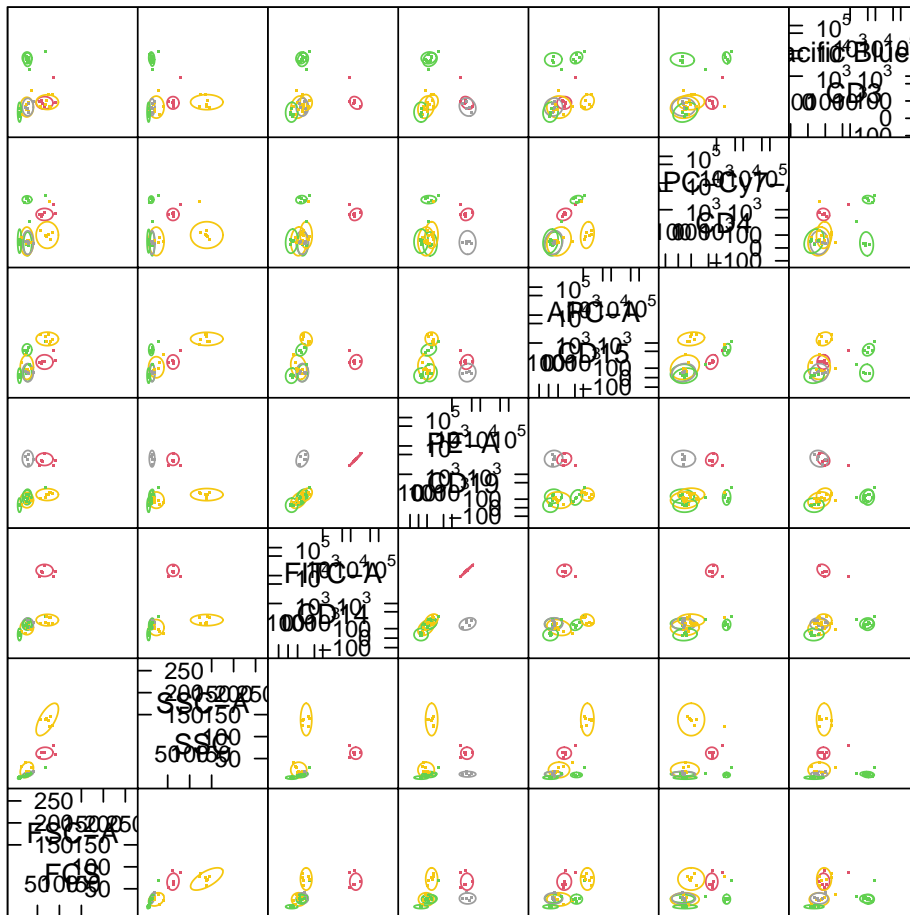
and pick the meta-clusters of the five commonly found population, with respect to the technical depletion to collect them in a first annotation level

```
> addLevel(meta,c(1),"leucocytes") <- c(1,2,6,7,10,14,18)
```

In the plot of this level the five major population are seen easily

```
> plot(meta, c(1))
```

1.all_leucocytes



and we identify the clusters for the particular populations successively by their expression levels.

```

> cls <- clusters(meta,c(1))
> sort(mu(meta,cls,7))      ## CD3 expression
[1] 0.5563285 1.0177510 1.0231479 1.4074683 1.4710931 5.3398778 5.5034995
> inc <- mu(meta,cls,7) > 5  ## CD3+ clusters
> cls[inc]
[1] 2 10
> mu(meta,cls[inc],6)      ## CD4 expression
[1] 0.3526607 4.1704618
> addLevel(meta,c(1,1), "CD3+CD4+") <- 10
> addLevel(meta,c(1,2), "CD3+CD4-") <- 2
> cls <- unclassified(meta,c(1))
> sort(mu(meta,cls,5))      ## CD15 expression
[1] 0.1607839 0.4098828 0.8552890 1.2885715 3.1912791
    
```

immunoClust

```

> inc <- mu(meta,cls,5) > 3
> addLevel(meta,c(1,3), "CD15+") <- cls[inc]
> cls <- unclassified(meta,c(1))
> sort(mu(meta,cls,3))      ## CD14 expression

[1] 0.2970245 0.8748380 1.1685025 5.5770927

> inc <- mu(meta,cls,3) > 5
> addLevel(meta,c(1,4), "CD14+") <- cls[inc]
> cls <- unclassified(meta,c(1))
> sort(mu(meta,cls,4))      ## CD19 expression

[1] 0.2053237 0.6140560 3.9759928

> inc <- mu(meta,cls,4) > 3
> addLevel(meta,c(1,5), "CD19+") <- cls[inc]

```

The whole analysis is performed on uncompensated FC data, thus the high CD19 values on the CD14-population is explained by spillover of FITC into PE.

The event numbers of each meta-cluster and each sample are extracted in a numeric matrix by the `meta.numEvents` function.

```

> tbl <- meta.numEvents(meta, out.all=FALSE)
> tbl[,1:5]

```

	12543	12546	12549	12552	12555
1.1.all_leucocytes_CD3+CD4+.10.green3	1107	3425	1585	0	0
1.2.all_leucocytes_CD3+CD4-.2.green3	389	1079	574	433	46
1.3.all_leucocytes_CD15+.14.yellow	6459	0	5717	7280	7417
1.4.all_leucocytes_CD14+.1.red	898	0	0	761	950
1.5.all_leucocytes_CD19+.7.gray	0	926	452	331	325
1.all_leucocytes.6.yellow	344	695	780	527	400
1.all_leucocytes.18.green3	151	495	247	247	278
.all.3.blue	50	0	0	62	94
.all.4.cyan	0	173	0	0	0
.all.5.magenta	0	102	0	0	0
.all.8.black	143	8	199	0	0
.all.9.red	71	145	0	0	0
.all.11.blue	0	220	0	0	0
.all.12.cyan	0	1447	0	0	0
.all.13.magenta	0	923	0	0	0
.all.15.gray	70	0	0	0	0
.all.16.black	0	24	0	0	0
.all.17.red	0	103	10	0	0
.all.19.blue	0	77	0	0	0
.all.20.cyan	0	0	0	95	0
.all.21.magenta	0	0	132	0	0
.all.22.yellow	0	0	40	0	0

Each row denotes an annotated hierarchical level or/and meta-cluster and each column a data sample used in meta-clustering. The row names give the annotated population name, the meta-cluster index and the default color used in the plot routines for each meta-cluster.

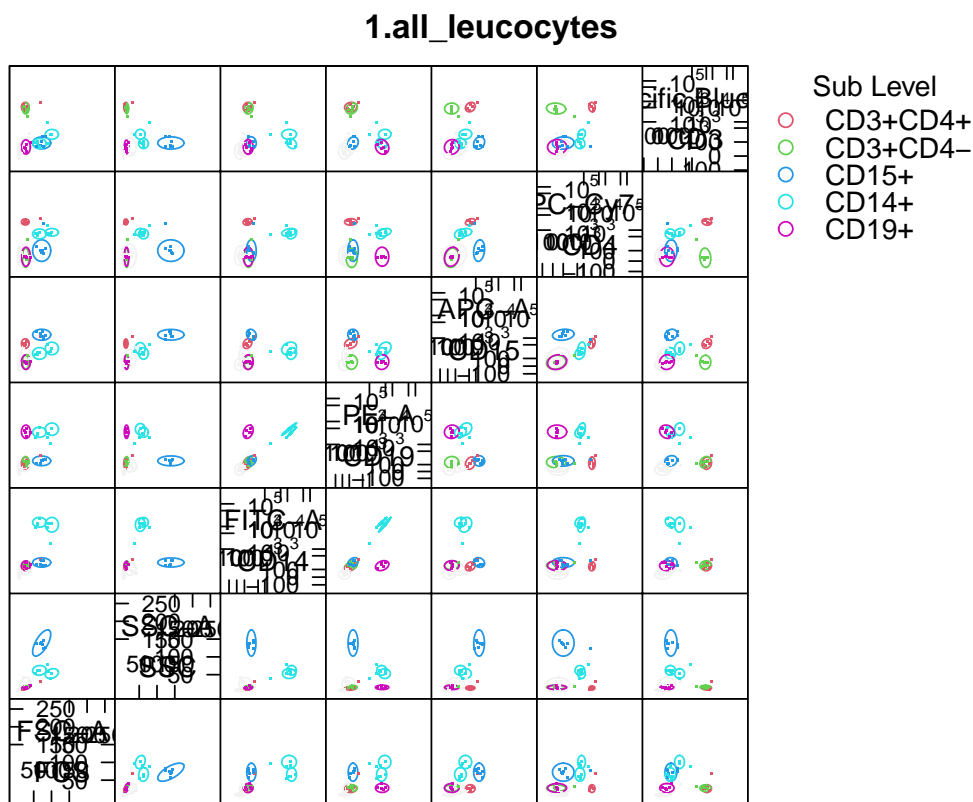
immunoClust

In the last columns additionally the meta-cluster centre values in each parameter are given, which helps to identify the meta-clusters. Further export functions retrieve relative cell event frequencies and sample meta-cluster centre values in a particular parameter.

We see here, that for sample 12546 where the CD15-cells are depleted, the CD14-population is missing. Anyway, this missing cluster could be in the so far unclassified clusters.

```
> move(meta,c(1,4)) <- 13
```

```
> plot(meta, c(1))
```



We see the CD14 population of sample 12546 shifted in FSC and CD3 expression levels, probably due to technical variation in the measurement of the CD15-depleted sample, where the granulocytes are missing which constitute about 60% - 70% of the events in the other samples.

5 Session Info

The documentation and example output was compiled and obtained on the system:

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

- R version 4.0.2 (2020-06-22), x86_64-pc-linux-gnu
- Locale: LC_CTYPE=en_US.UTF-8, LC_NUMERIC=C, LC_TIME=en_US.UTF-8, LC_COLLATE=C, LC_MONETARY=en_US.UTF-8, LC_MESSAGES=en_US.UTF-8, LC_PAPER=en_US.UTF-8, LC_NAME=C, LC_ADDRESS=C, LC_TELEPHONE=C, LC_MEASUREMENT=en_US.UTF-8, LC_IDENTIFICATION=C
- Running under: Ubuntu 18.04.4 LTS
- Matrix products: default
- BLAS: /home/biocbuild/bbs-3.11-bioc/R/lib/libRblas.so
- LAPACK: /home/biocbuild/bbs-3.11-bioc/R/lib/libRlapack.so
- Base packages: base, datasets, grDevices, graphics, grid, methods, stats, utils
- Other packages: flowCore 2.0.1, immunoClust 1.20.1, lattice 0.20-41
- Loaded via a namespace (and not attached): Biobase 2.48.0, BiocGenerics 0.34.0, BiocManager 1.30.10, BiocStyle 2.16.0, RProtoBufLib 2.0.0, Rcpp 1.0.5, RcppParallel 5.0.2, compiler 4.0.2, cytolib 2.0.3, digest 0.6.25, evaluate 0.14, htmltools 0.5.0, knitr 1.29, matrixStats 0.56.0, parallel 4.0.2, rlang 0.4.7, rmarkdown 2.3, stats4 4.0.2, tools 4.0.2, xfun 0.16, yaml 2.2.1