Using self-organizing maps for visualization and interpretation of cytometry data

Sofie Van Gassen, Britt Callebaut and Yvan Saeys

Ghent University

September, 2014

Abstract

The *FlowSOM* package provides new visualization opportunities for cytometry data. A four-step algorithm is provided: first, the data is read and preprocessed, then a self-organizing map is trained and a minimal spanning tree is build, and finally, a meta-clustering is computed. Several plotting options are available, using star charts to visualize marker intensities and pie charts to visualize correspondence with manual gating results or other automatic clustering results.

1. Reading the data

The FlowSOM package has several input options.

The first possibility is to use an array of character strings, specifying paths to files or directories. When given a path to a directory, all files in the directory will be considered. This process does not happen recursively. You can specify a pattern to use only a selection of the files. The default pattern is `".fcs"`, making sure that only fcs-files are selected. When you are already working with your data in *R*, it might be easier to use a *flowFrame* or *flowSet* from the *flowCore* package as input. This is also supported. If multiple paths or a *flowSet* are provided, all data will be concatenated.

When reading the data, several pre-processing options are available. The data can be automatically compensated using a specified matrix, or using the `$SPILL` variable from the fcs-file. The data can be logicle transformed for specified columns. If no columns are provided, all columns from the spillover matrix will be transformed. Finally, the data can be scaled. By default, it will scale to a mean of zero and standard deviation of one. However, specific scaling parameters can be set (see the base *R* `scale` function for more detail).

```
> set.seed(42)
> library(flowCore)
> library(FlowSOM)
> fileName <- system.file("extdata","lymphocytes.fcs",
+                         package="FlowSOM")
> fSOM <- ReadInput(fileName,compensate = TRUE,transform = TRUE,
+                       toTransform=c(8:18),scale = TRUE)
> ff <- read.FCS(fileName)
> fSOM <- ReadInput(ff,compensate = TRUE,transform = TRUE, scale = TRUE)
```

This function returns a FlowSOM object, which is actually a list containing several parameters. The data is stored as a matrix in $data, and all parameter settings to read the data are also stored. The begin and end indices of the subsets from the different files can be found in $metadata.

```
> str(fSOM)
```

```
List of 11
 $ pattern       : chr ".fcs"
 $ compensate    : logi TRUE
 $ spillover     : num [1:11, 1:11] 1.00 4.84e-04 8.11e-04 8.63e-05 4.35e-04 ...
  ..- attr(*, "dimnames")=List of 2
  .. ..$ : NULL
```

```
  .. ..$ : chr [1:11] "FITC-A" "Pacific Blue-A" "AmCyan-A" "Qdot 605-A" ...
 $ transform     : logi TRUE
 $ toTransform   : chr [1:11] "FITC-A" "Pacific Blue-A" "AmCyan-A" "Qdot 605-A" ...
 $ scale         : logi TRUE
 $ prettyColnames: Named chr [1:18] "Time <Time>" "FSC-A <FSC-A>" "FSC-H <FSC-H>" "FSC-W <FSC-W>" ...
  ..- attr(*, "names")= chr [1:18] "Time" "FSC-A" "FSC-H" "FSC-W" ...
 $ data          : num [1:19225, 1:18] -1.65 -1.65 -1.65 -1.65 -1.65 ...
  ..- attr(*, "dimnames")=List of 2
  .. ..$ : NULL
  .. ..$ : Named chr [1:18] "Time" "FSC-A" "FSC-H" "FSC-W" ...
  .. .. ..- attr(*, "names")= chr [1:18] "$P1N" "$P2N" "$P3N" "$P4N" ...
 $ metaData      :List of 1
  ..$ /tmp/Rtmpyb9Km5/Rinst4bf0348a09b/FlowSOM/extdata/lymphocytes.fcs: num [1:2] 1 19225
 $ scaled.center : Named num [1:18] 3356 88594 68698 84405 36886 ...
  ..- attr(*, "names")= chr [1:18] "Time" "FSC-A" "FSC-H" "FSC-W" ...
 $ scaled.scale  : Named num [1:18] 2038 15064 3236 12997 13923 ...
  ..- attr(*, "names")= chr [1:18] "Time" "FSC-A" "FSC-H" "FSC-W" ...
 - attr(*, "class")= chr "FlowSOM"
```

2. Building the self-organizing map

The next step in the algorithm is to build a self-organizing map. Several parameters for the self-organizing map algorithm can be provided, such as the dimensions of the grid, the learning rate, the number of times the dataset has to be presented. However, the most important parameter to decide is on which columns the self-organizing map should be trained. This should contain all the parameters that are useful to identify cell types, and exclude parameters of which you want to study the behaviour on all cell types such as activation markers.

The BuildSOM function expects a FlowSOM object as input, and will return a FlowSOM object with all information about the self organizing map added in the map parameter of the FlowSOM object.

```
> fSOM <- BuildSOM(fSOM,colsToUse = c(9,12,14:18))
> str(fSOM$map)
```

```
List of 13
 $ xdim      : num 10
 $ ydim      : num 10
 $ rlen      : num 10
 $ mst       : num 1
 $ alpha     :List of 1
  ..$ : num [1:2] 0.05 0.01
 $ radius    :List of 1
  ..$ : num [1:2] 6 0
 $ init      : logi FALSE
 $ distf     : num 2
 $ grid      :'data.frame':        100 obs. of  2 variables:
  ..$ Var1: int [1:100] 1 2 3 4 5 6 7 8 9 10 ...
  ..$ Var2: int [1:100] 1 1 1 1 1 1 1 1 1 1 ...
  ..- attr(*, "out.attrs")=List of 2
  .. ..$ dim     : int [1:2] 10 10
  .. ..$ dimnames:List of 2
  .. .. ..$ Var1: chr [1:10] "Var1= 1" "Var1= 2" "Var1= 3" "Var1= 4" ...
  .. .. ..$ Var2: chr [1:10] "Var2= 1" "Var2= 2" "Var2= 3" "Var2= 4" ...
 $ codes     : num [1:100, 1:7] 2.729 2.854 1.973 -0.052 -0.897 ...
  ..- attr(*, "dimnames")=List of 2
  .. ..$ : NULL
```

```
  .. ..$ : chr [1:7] "Pacific Blue-A" "APC-A" "APC-Cy7-A" "PE-A" ...
 $ mapping   : num [1:19225, 1:2] 10 99 95 65 62 11 1 61 11 37 ...
 $ colsUsed  : num [1:7] 9 12 14 15 16 17 18
 $ meanValues:'data.frame':        100 obs. of  18 variables:
  ..$ Time             : num [1:100] -0.473 0.39 -0.0288 0.1469 0.2822 ...
  ..$ FSC-A            : num [1:100] -0.0514 0.1362 1.5226 0.2978 0.208 ...
  ..$ FSC-H            : num [1:100] -0.136 -0.0484 3.9556 0.5586 0.3191 ...
  ..$ FSC-W            : num [1:100] -0.0125 0.1702 0.3893 0.1578 0.1245 ...
  ..$ SSC-A            : num [1:100] -0.4589 -0.1547 1.1836 0.0403 0.012 ...
  ..$ SSC-H            : num [1:100] -0.267 -0.023 3.286 0.605 0.265 ...
  ..$ SSC-W            : num [1:100] -0.442 -0.154 0.6431 -0.0547 -0.0311 ...
  ..$ FITC-A           : num [1:100] 0.0808 0.0473 0.624 -0.0616 -0.3252 ...
  ..$ Pacific Blue-A   : num [1:100] 2.699 2.869 2.426 -0.043 -0.894 ...
  ..$ AmCyan-A         : num [1:100] 0.142 0.143 0.713 -0.326 -0.258 ...
  ..$ Qdot 605-A       : num [1:100] -0.1146 -0.1156 -0.0138 -0.2979 -0.5447 ...
  ..$ APC-A            : num [1:100] -0.683 -0.664 -0.495 -1.217 -0.977 ...
  ..$ Alexa Fluor 700-A: num [1:100] 0.751 0.956 1.128 0.479 0.661 ...
  ..$ APC-Cy7-A        : num [1:100] 0.633 0.747 1.002 0.981 1.084 ...
  ..$ PE-A             : num [1:100] -0.927 0.475 0.529 0.61 0.786 ...
  ..$ PE-Texas Red-A   : num [1:100] -0.814 -1 0.997 1.125 1.167 ...
  ..$ PE-Cy5-A         : num [1:100] -0.171 -0.108 0.117 0.761 0.575 ...
  ..$ PE-Cy7-A         : num [1:100] 0.557 0.625 0.874 0.834 0.973 ...
```

3. Building the minimal spanning tree

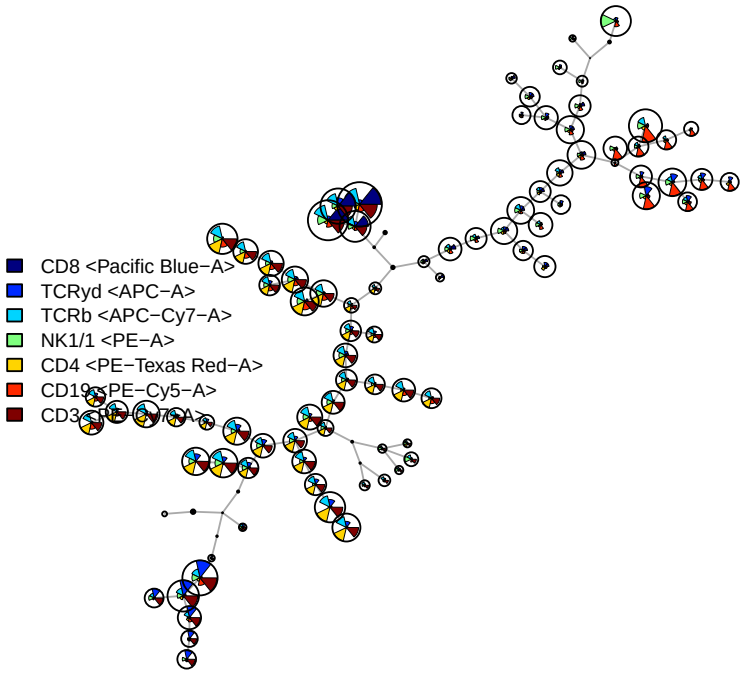
The third step of FlowSOM is to build the minimal spanning tree. This will again return a FlowSOM object, with extra information contained in the $MST parameter.

```
> fSOM <- BuildMST(fSOM)
> str(fSOM$MST)
```

```
List of 3
 $ graph:IGRAPH UNW- 100 99 --
+ attr: name (v/c), weight (e/n)
+ edges (vertex names):
 [1] 1 --12  2 --12  3 --13  4 --15  5 --6   6 --16  7 --17  8 --9   9 --18  9 --19
[11] 10--20  11--12  12--13  13--23  14--15  14--16  14--26  15--25  17--28  19--29
[21] 20--30  21--22  22--23  22--32  23--24  24--25  25--36  27--36  28--38  29--40
[31] 30--39  31--63  32--54  33--42  33--71  34--35  35--45  35--46  36--58  37--47
[41] 38--48  38--58  39--49  40--50  41--51  41--53  42--44  42--52  43--44  44--53
[51] 46--47  46--56  47--48  47--49  49--60  50--60  53--54  55--65  55--75  56--57
[61] 56--66  56--67  59--69  59--70  60--69  61--62  62--63  62--72  64--74  68--69
[71] 68--78  71--72  71--83  72--73  73--74  74--75  75--76  77--78  78--79  78--87
[81] 79--80  81--83  81--91  81--92  82--83  83--84  84--85  84--96  86--96  87--97
[91] 88--89  88--97  89--90  89--98  90--100 92--93  94--95  95--96  99--100
 $ l    : num [1:100, 1:2] -2.24 -3.49 -1.23 -4.41 -7.67 ...
 $ size : num [1:100] 14 12.6 1.1 8.79 9.52 ...
```

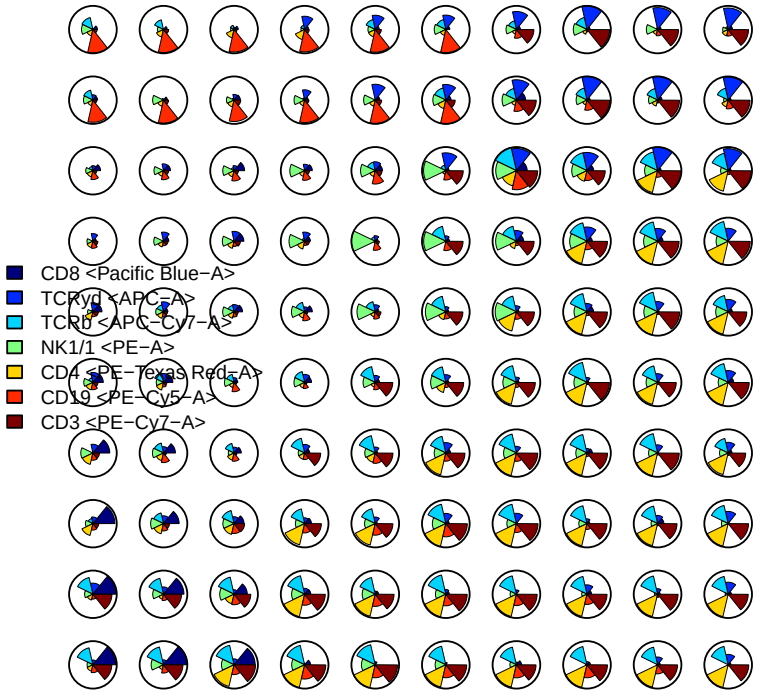
Once this step is finished, the FlowSOM object can be used for visualization.

```
> PlotStars(fSOM)
```



If you do not want the size to depend on the number of cells assigned to a node, you can reset the node size.

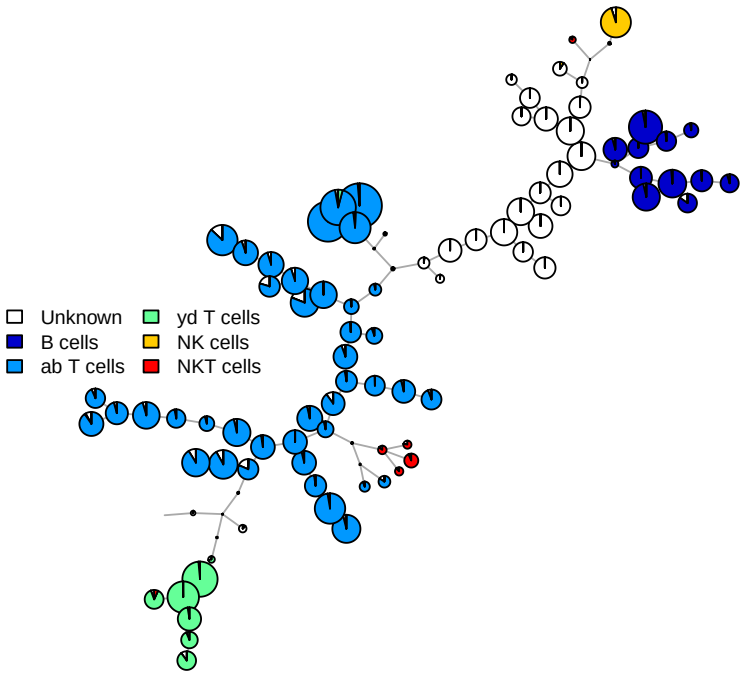
```
> fSOM <- UpdateNodeSize(fSOM, reset=TRUE)
> PlotStars(fSOM,MST=FALSE)
> fSOM <- UpdateNodeSize(fSOM)
```



It might also be interesting to compare with a manual gating.

```
> library(flowUtils)
> flowEnv <- new.env()
> ff_c <- compensate(ff,ff@description$SPILL)
> colnames(ff_c)[8:18] <- paste("Comp-",colnames(ff_c)[8:18],sep="")
> gatingFile <- system.file("extdata","manualGating.xml",
+                           package="FlowSOM")
> read.gatingML(gatingFile, flowEnv)
> filterList <- list( "B cells" = flowEnv$ID52300206,
+                     "ab T cells" = flowEnv$ID785879196,
+                     "yd T cells" = flowEnv$ID188379411,
+                     "NK cells" = flowEnv$ID1229333490,
+                     "NKT cells" = flowEnv$ID275096433
+                   )
> results <- list()
> for(cellType in names(filterList)){
+     results[[cellType]] <- filter(ff_c,filterList[[cellType]])@subSet
+ }
> manual <- rep("Unknown",nrow(ff))
> for(celltype in names(results)){
```

```
+     manual[results[[celltype]]] <- celltype
+ }
> # Use a factor to define order of the cell types
> manual <- factor(manual,levels = c("Unknown","B cells",
+                                    "ab T cells","yd T cells",
+                                    "NK cells","NKT cells"))
> PlotPies(fSOM,cellTypes=manual)
```

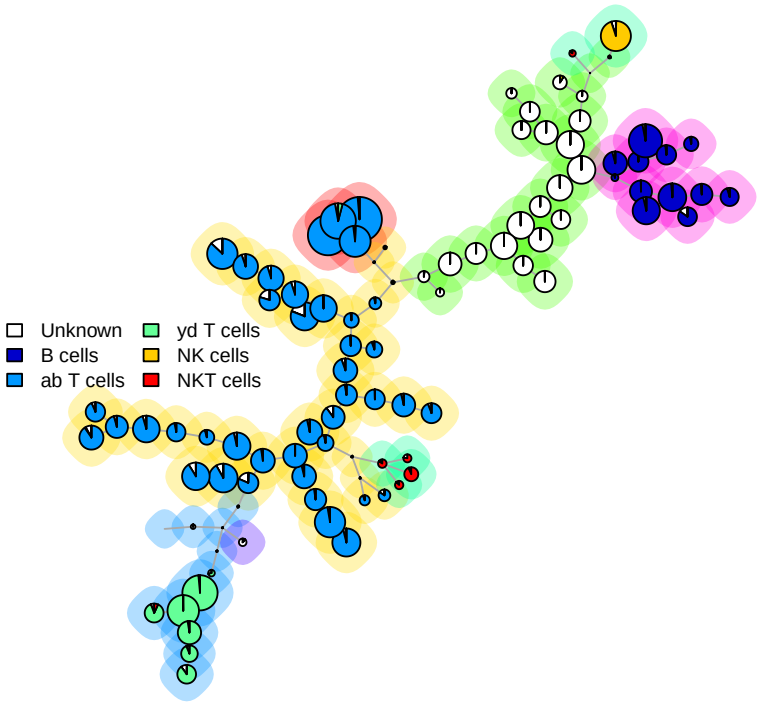


4. Metaclustering the data

The fourth step of the FlowSOM algorithm is to perform a meta-clustering of the data. This can be the first step in further analysis of the data, and often gives a good approximation of manual gating results.

If you have background knowledge about the number of cell types you are looking for, it might be optimal to provide this number to the algorithm.

```
> metaClustering <- metaClustering_consensus(fSOM$map$codes,k=7)
> PlotPies(fSOM,cellTypes=manual,clusters = metaClustering)
```



You can also extract the metaClustering for each cell individually

```
> metaClustering_perCell <- metaClustering[fSOM$map$mapping[,1]]
```

5. Summary

In summary, the FlowSOM package provides some new ways to look at cytometry data. It can help to keep an overview of how all markers are behaving on different cell types, and to reduce the probability of overlooking interesting things that are present in the data.