Introduction to FlowSOM analyse

Anja Bråthen Kristoffersen

last updated

date()

## [1] "Tue May 24 13:55:53 2022"

This document describes the different lines of the file “FlowSOM\_analyse.R”. To perform your own flowSOM analysis save the file “FlowSOM\_analyse.R” from github <https://github.com/folkehelseinstituttet/cytof> give it a new name and change the necessary paths and values throughout the document to use it on your own data. The file “FlowSOM\_analyse.R” is sourced from the file “runFlowSOM\_analyse.R” which also can be found on github together with an explanatory file called “Introduction to runFlowSOM\_analyse.R”. These documents only make a clustering of your data and some additional files that later will be used when analysing the flowSOM data together with metadata, examples of analysing of the clustering data together with metadata can be found in resultater\_Regression\_clusters.Rmd which is sourced from runResults\_Regression\_Clusters.R.

This file is as explained sourced from “runFlowSOM\_analyse.R”. In “runFlowSOM\_analyse.R” the variable “selectedEvents”, “fcs\_data”, “file\_names”, “posNeg”, “channel”, “n\_per\_file”, “seed”, “kanalnavn”, “kanaler”, “xdim”, “ydim”, “ks”, “utSti”, “o”, “column\_cluster” and “tamed” are already defined. In brief

* selectedEvents is either TRUE or FALSE indicating if just some events or all events are used.
* fcs\_data is your cytof data, as a list of matrixes per file, you get this data by running the file readDataToAnalyse.R
* file\_names is the names of the files in fcs\_data in the same order as the list.
* posNeg is a list of pregated matrix with values 0, 1, 2. posNeg has the same size as fcs\_data but each event for each channel has one of the values 0 (negative), 1 (low or positive), 2 (high, if used), only applicable if selectedEvents is TRUE.
* channel indicate which channel in posNeg that will be used, only applicable if selectedEvents is TRUE.
* n\_per\_file is a value that indicate how many cells that should be randomly chosen.
* seed is a number that ensure that you get the same clustering if the file is run again.
* kanalnavn is a vector with names of the channels that will be used in the clustering. These names correspond to the names in the cytof dataset.
* kanaler is a vector of the same length as kanalnavn, but this vector has names that correspond to the names in kanalnavn, but they are easier to remember.
* xdim and ydim are two variables indication how many nodes that will be made in the pre clustering done by flowSOM. When less than 30 cluster are expected there could be enough with 100 nodes (which give xdim = ydim = 10). While for more clusters more nodes are needed here 14\*14 = 196 nodes are used.
* ks is a vector of number of clusters that you want to make of the clustering with this seed.
* utSti which is the path to the folder where you want to save the clustering result.
* o is optional, but must be included if you want a different order of your channels in the heatmap.
* tamed a list of which filenames to use in the clustering

First the random events to use in the clustering are chosen. By using the R function set.seed() it is ensured that the same random cells will be chosen every time the script is run. If selectedEvenst is TRUE the function random\_events\_from\_selected\_events() is used. This function uses the variable posNeg and a given marker here with the variable name channel to randomly select n\_per\_file number of cells that all have the value 1 (or 2) for channel in the posNeg list of matrixes. If selectedEvenst is FALSE the random events are chosen from all events in each matrix. To find the number of events in each file the function number\_of\_events() is used and then n\_per\_file number of events are chosen by using the function random\_events()

if(selectedEvents == T){  
 set.seed(seed)  
 random\_events\_data <- random\_events\_from\_selected\_events(posNeg = posNeg, marker = channel, n = n\_per\_file)  
 ext\_name <- channel  
} else {  
 set.seed(seed)  
 number\_of\_events\_data <- number\_of\_events(data = fcs\_data, file\_names = file\_names)  
 random\_events\_data <- random\_events(number\_of\_events\_data, n = n\_per\_file)  
 ext\_name <- "all"  
}

Now when the random events to use are chosen a matrix with the chosen events for each file is made by using the function list\_to\_matrix\_selected\_events(). This function takes the cytof data stored in fcs\_data, random events chosen stored in random\_events\_data, the file\_names of the fcs\_data, which channels to use for gating, stored in kanalnavn in addtion to archSin a boolean variable with default TRUE, cofactor which have default 5 and scale which have default FALSE

arcSindataMatrix <- list\_to\_matrix\_selected\_events(data = fcs\_data,   
 kept\_events = random\_events\_data,   
 file\_names = file\_names,   
 channels = kanalnavn,  
 archSin = T,   
 cofactor = 5,   
 scale = F  
)  
colnames(arcSindataMatrix)[1:length(kanaler)] <- kanaler  
print("arcsin")

then all parameters are stored together in a list called params

params <- list(seed = seed,  
 data = arcSindataMatrix[arcSindataMatrix$dataset %in% tamed, ],  
 n\_random\_for\_plotting\_per\_fil = 5000,  
 kanaler = kanaler,  
 scaling = TRUE,  
 column\_cluster = column\_cluster,  
 o = NULL,  
 ydim = xdim,   
 xdim = ydim  
)  
  
if(is.null(params$o)){  
 params$o <- params$kanaler  
}

The seed is again set to make sure the FlowSOM analysis are done similarly each time repeated for the same seed. And the initial FlowSOM analysis are run assigning each event in the arcSindataMatrix to a node in the xdim times ydim node network.

set.seed(params$seed) # to ensure same plot every time  
out <- FlowSOM::ReadInput(as.matrix(params$data[,params$kanaler]), transform = F, scale = params$scaling)  
print(1)  
out <- FlowSOM::BuildSOM(out, colsToUse = 1:(ncol(params$data[,params$kanaler])), xdim = params$xdim, ydim = params$ydim)  
print(2)  
out <- FlowSOM::BuildMST(out)  
print(3)  
cluster\_FlowSOM\_pre <- out$map$mapping[, 1]

Then for each value k in the vector ks the clustering of nodes to k clusters are done, the 10 and 90 % quantiles together with the median signal of all events in the clusters are calculated and written to a file in the folder utSti for every channel used. A heatmap is calculated and saved. The total number of events in each cluster is calculated and saved to a file. The total number of events per file is also calculated and saved.

for(k in ks){  
 print(k)  
 set.seed(params$seed)  
 out\_k <- FlowSOM::metaClustering\_consensus(out$map$codes, k = k, seed = params$seed)  
 cluster\_FlowSOM\_k <- out\_k[cluster\_FlowSOM\_pre]  
 cluster\_FlowSOM\_k\_factor <- factor(cluster\_FlowSOM\_k, levels = 1:k)  
print("cluster")  
 q5\_k <- q\_per\_cluster\_marker(data = params$data[,params$kanaler], kluster = cluster\_FlowSOM\_k, probs = 0.05)  
 write.csv2(q5\_k, fs::path(utSti, paste0("q5\_per\_kluster\_k\_", k, "\_seed", params$seed, ext\_name, ".csv")))  
 q10\_k <- q\_per\_cluster\_marker(data = params$data[,params$kanaler], kluster = cluster\_FlowSOM\_k, probs = 0.1)  
 write.csv2(q10\_k, fs::path(utSti, paste0("q10\_per\_kluster\_k\_", k, "\_seed", params$seed, ext\_name, ".csv")))  
 q25\_k <- q\_per\_cluster\_marker(data = params$data[,params$kanaler], kluster = cluster\_FlowSOM\_k, probs = 0.25)  
 write.csv2(q25\_k, fs::path(utSti, paste0("25\_per\_kluster\_k\_", k, "\_seed", params$seed, ext\_name, ".csv")))  
 q75\_k <- q\_per\_cluster\_marker(data = params$data[,params$kanaler], kluster = cluster\_FlowSOM\_k, probs = 0.75)  
 write.csv2(q75\_k, fs::path(utSti, paste0("q75\_per\_kluster\_k\_", k, "\_seed", params$seed, ext\_name, ".csv")))  
 q90\_k <- q\_per\_cluster\_marker(data = params$data[,params$kanaler], kluster = cluster\_FlowSOM\_k, probs = 0.9)  
 write.csv2(q90\_k, fs::path(utSti, paste0("q90\_per\_kluster\_k\_", k, "\_seed", params$seed, ext\_name, ".csv")))  
 q95\_k <- q\_per\_cluster\_marker(data = params$data[,params$kanaler], kluster = cluster\_FlowSOM\_k, probs = 0.95)  
 write.csv2(q95\_k, fs::path(utSti, paste0("q95\_per\_kuster\_k\_", k, "\_seed", params$seed, ext\_name, ".csv")))  
print("q")  
  
 medians\_k <- median\_per\_cluster\_marker(data = params$data[,params$kanaler], kluster = cluster\_FlowSOM\_k)  
 write.csv2(medians\_k, fs::path(utSti, paste0("medians\_per\_kluster\_k\_", k, "\_seed", params$seed, ext\_name, ".csv")))  
print("median")  
  
  
  
 tiff(fs::path(utSti, paste0("heatmap\_median\_k\_", k, "\_cluster\_seed", seed, ext\_name, ".tiff")), width = 1000, height = 800)  
 Heatmap(as.matrix(medians\_k[,params$o], cluster\_columns = params$column\_cluster))  
 dev.off()  
print("heatmap")  
  
  
 totalt\_antall\_i\_kluster <- table(cluster\_FlowSOM\_k\_factor)  
   
  
 antallPerKluster <- cbind(totalt\_antall\_i\_kluster, totalt\_antall\_i\_kluster/nrow(params$data))  
 colnames(antallPerKluster) <- c("antall", "prosent")  
 write.csv2(antallPerKluster, fs::path(utSti, paste0("antallPerCluster\_k\_", k, "\_seed", params$seed, ext\_name, ".csv")))  
print("totalt antall")  
  
   
 perSample <- table(params$data$dataset, cluster\_FlowSOM\_k\_factor)  
 colnames(perSample) <- paste0("kluster\_", 1:k)  
  
 write.csv2(perSample, fs::path(utSti, paste0("antallPerClusterOgPat\_k\_", k, "\_seed", params$seed, ext\_name, ".csv")))  
print("antall per fil")  
  
}