Project Almaso Recap

Autors:

ETC…

The goal of the project is to build a pipeline to analyse flow cytometry data in the manner of scRNA-Seq data analysis. This present document will list all choices made when developing the workflow (libraries used, pre-processing methods, number of axes chosen…). The presentation of the final pipeline and the explanation for each function created is to be found in the vignette attached.

Table des matières

[The Data 1](#_Toc126005780)

[Pre-processing 1](#_Toc126005781)

[Dimensionality reduction 3](#_Toc126005782)

[Clustering 3](#_Toc126005783)

[Major adjustments 5](#_Toc126005784)

[Differential Expression 7](#_Toc126005785)

[Biological Interpretation 7](#_Toc126005786)

[Discussion 7](#_Toc126005787)

[Conclusion 9](#_Toc126005788)

# The Data

The datasets on which we work are from mice with and without tumours. We had flow cytometry datasets and equivalent scRNA-Seq datasets. The aim is to conduct an analysis on both cytometry data and scRNA-Seq data, to highlight similarities and differences. As cytometry data are usually analysed through gating techniques and not unsupervised learning, our work was more focused on cytometry data analysis.

As a reminder, flow cytometry is a technique that analyses the expression of cell surface and intracellular molecules. The fluorescent intensity produced by fluorescent-labelled antibodies detecting protein are measured as cells flow one by one in front of lasers. Different fluorescent markers can be used simultaneously to identify different specific proteins. 2 types of parameters can be measured in flow cytometry:

* Structural parameters: forward scatter (FSC), which detects cell size, and side scatter, that provides information about the internal complexity of a cell. This information is usually used in the first steps of the gating process to discriminate by cell size.
* Fluorescence markers: intensity measurement gives expression level of specific protein stained by a fluorescent marker.

Our dataset is composed by a matrix of expression of 19 parameters (6 structural and 13 markers), stored in columns, and thousands of cells (stored in lines). We decided, as structural parameters and fluorescent markers do not convey the same type of information, that we will not use them both for clustering. We will focus mostly on the fluorescent markers.

# Pre-processing

The first step of our pipeline is to pre-process the data in order to clean them for further analysis. Our pre-processing pipeline contains 3 steps: compensation, quality control and normalisation.

* **Compensation**

This is a necessary processing step for flow cytometry data. During the fluorescence detection, some detectors may register fluorochrome signals that do not correspond to their dye. So, to compensate for this spillover, a spillover matrix (that expresses background noise and secondary signals for each channel) is applied the expression matrix (as a linear combination). So the general idea is to compensate the spillover phenomenon between channels by applying a specific matrix.

A lot of packages contain functions that does it. We choose the *flowCore* function. At first *compensate()* did not work because the data received was already compensated and did not have the spillover matrix anymore. But we used the function with a non-compensated dataset, and it worked fine.

* **Quality control**

The goal of quality control (QC) is to remove outliers and anomalies. This step is as necessary in flow cytometry data analysis as in scRNA-Seq data analysis. Since the sources of outliers are very different between flow cytometry and scRNA-Seq, we could not use scRNA-Seq QC packages.

Quality control functions looks for anomalies and peaks that show:

* Instability of flow rate=> remove flowrates that deviate from the median.
* Instability of signal acquisition => remove intensity that deviate from the mean.
* Inconsistency in the signal dynamic range => remove outliers by Z-score and negatives values.

After comparing 3 different packages (*flowAI*, *flowClean* and *peacoQC*), we first decided to use *flowAI* because it is one of the most used in flow cytometry analysis and it seemed to give more satisfactory results for the clustering. However, it was not always working on all computers because of version conflicts.

We then choose the *peacoQC* package. It gives good results.

|  |  |  |  |
| --- | --- | --- | --- |
| **Sample** | **Cells counts before  preprocessing** | **Cells counts after preprocessing** | **% of cells removed** |
| VP4\_Tumor\_CD45+ cells.fcs | 205,726 | 178,726 | 13,12 % |
| Vp6\_Control\_CD45+ cells.fcs | 64,213 | 58,963 | 8,18 % |
| MixC\_tumeur\_CD45+ cells.fcs | 66,816 | 62,566 | 6,36 % |
| CD45pos2\_control\_CD45+.fcs | 69,421 | 67,171 | 3,24 % |

Table 1: quality control results with peaco\_QC

* **Normalization**

For this last step of pre-processing, we decided to treat the flow cytometry data as scRNA-Seq data. So, we decided to use a logCPM normalization method on the expression matrix.

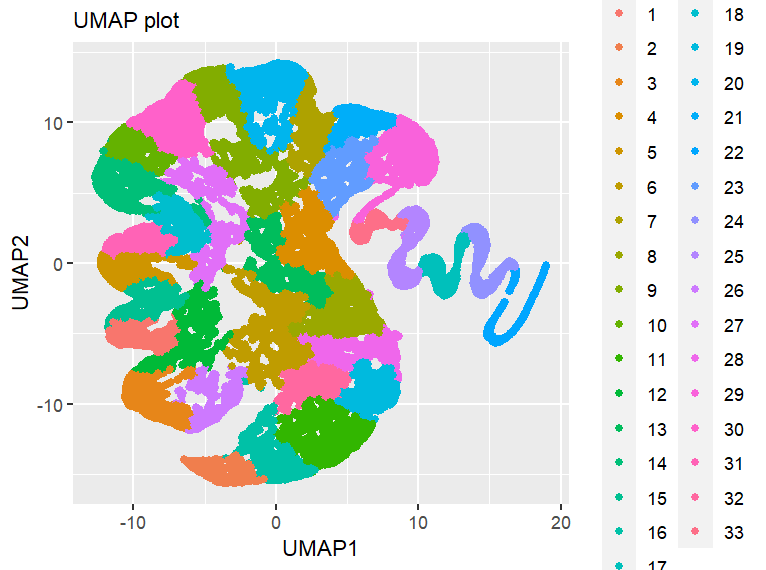
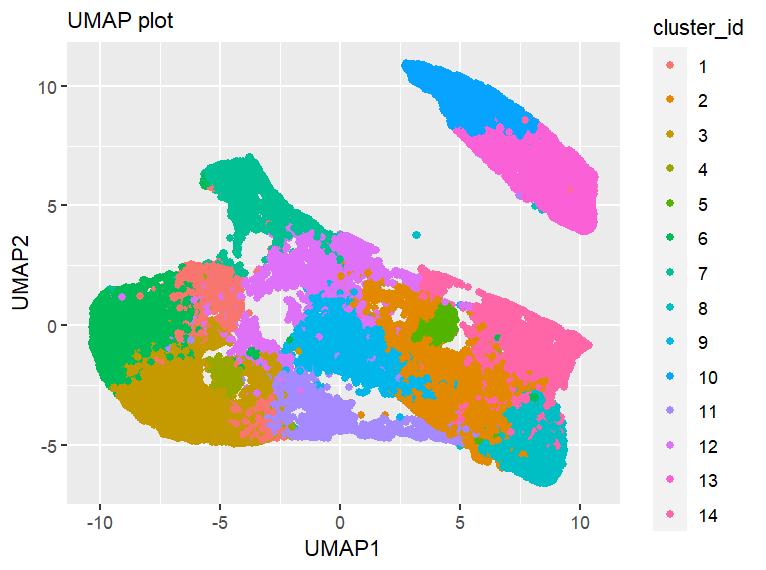
We did not integrate the log or arcsinh transformation step that is usual in flow cytometry pipelines because a log transformation is included in our normalization step.

# Dimensionality reduction

Since our datasets contained only 19 parameters, dimensionality reduction was not evidence. We therefore decided to run the pipeline with and without PCA and to compare the result. We observed that when using PCA, our clusters were better separated than without it.

We concluded that using PCA tends to increase the number of clusters (especially when using only 2 dimensions) but that the separation between clusters was increased.

Figure 1: UMAP representation on first 2 PCA dimensions (left). UMAP representation without PCA (right) . on the same dataset *cd45\_pos2\_control*



Our recommendation would therefore be to do a PCA, to increase the number of PC dimensions used for clustering to reduce the number of clusters, and to lower the Louvain clustering resolution to increase even more the separation between clusters.

# Clustering

To have a similar approach to what is usually done on scRNA-Seq data, we decided to do an UMAP representation. Then, we tested several clustering methods:

* **SNN + Louvain clustering,** which is the clustering method used in Seurat for scRNA-Seq data.
* **Cytosplore**, which is an app using HSNE-based Gaussian Mean Shift clustering to automatically compute flow cytometry clustering.
* **FlowSOM clustering,** which is a flow cytometry clustering package that seemed very popular in the literature.

We first tested Cytosplore which is an app that will calculate H-SNE clustering representation from cytometry data and output FCS files containing the clustering, that you can then visualize in RStudio. The advantage of this method is that it automatically calculates the clusters, but the issue is that this app cannot be used on a computing cluster. Moreover, Cytosplore is based on a R package called *cytofast*, which is no longer updated in the last version of Bioconductor. We therefore decided to drop Cytosplore.

Une image contenant texte

Description générée automatiquement

Figure 2: Cytosplore results (use HSNE)

We then compared the clustering results from **SNN + Louvain** and **FlowSOM** clusterings. We observed that with FlowSOM, most of the cells were always contained in one very big cluster, no matter the pre-processing or the dimensionality reduction. On the other hand, we had a reasonable number of clusters with a good separation with the SNN + Louvain clustering. Based on those observations we decided to only include the SNN + Louvain clustering in our pipeline. This method also has the advantage of being used for scRNA-Seq data. So, it also met the goal of treating cytometry data as scRNA-Seq data.

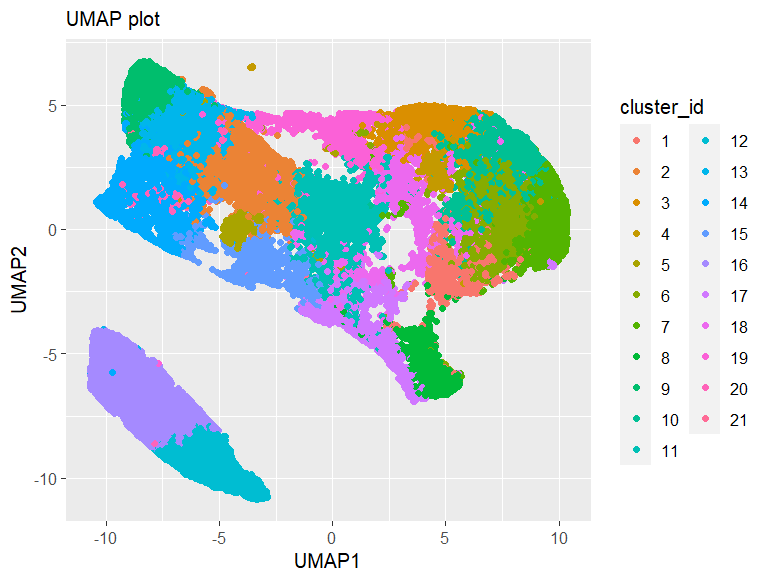
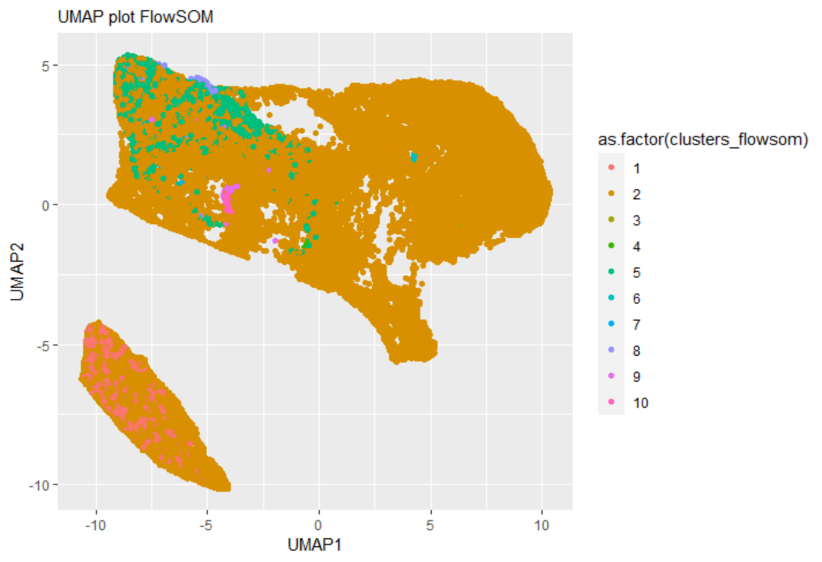


Figure 3: UMAP representation of FlowSOM clustering (left) and SNN+ Louvain clustering (right)

# Major adjustments

After running all the precedent steps on the *Vp6\_Control\_CD45+ cells.fcs* file, we observe an odd mechanism. When plotting the PCA, we saw that the first axis accounts for 91% of the variance. Then when taking 2 axes for PCA, we observe an UMAP with a potato shape.

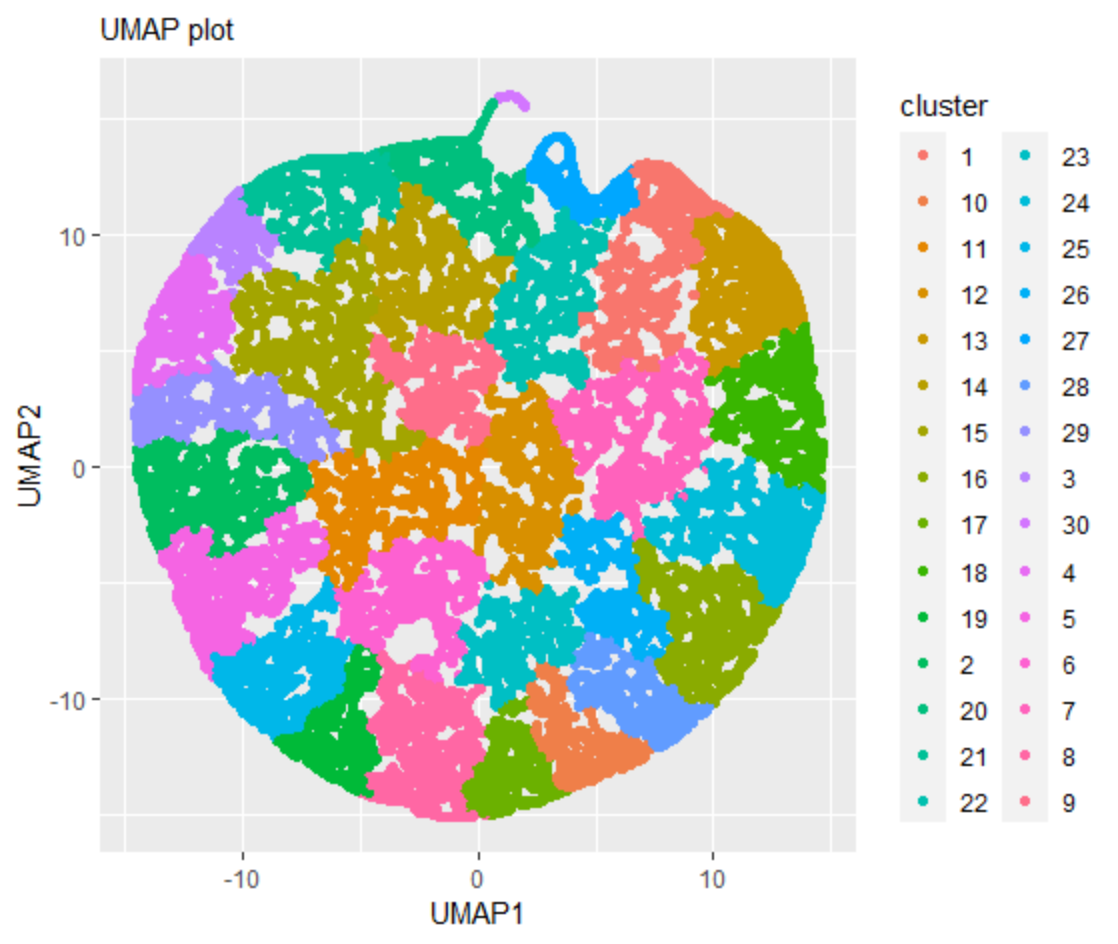
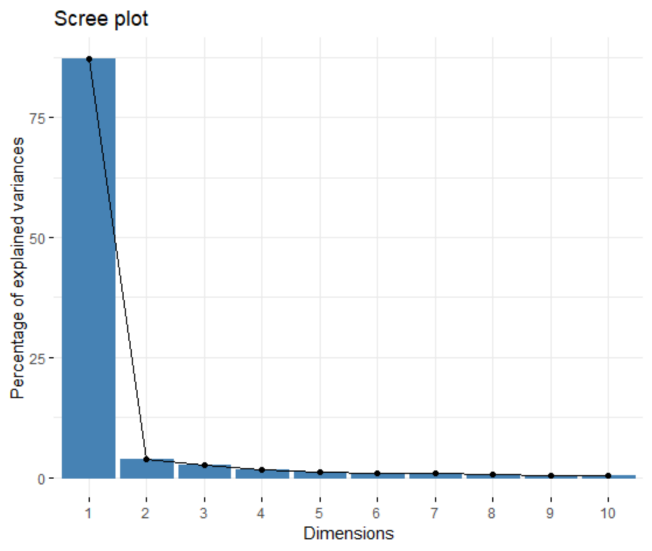
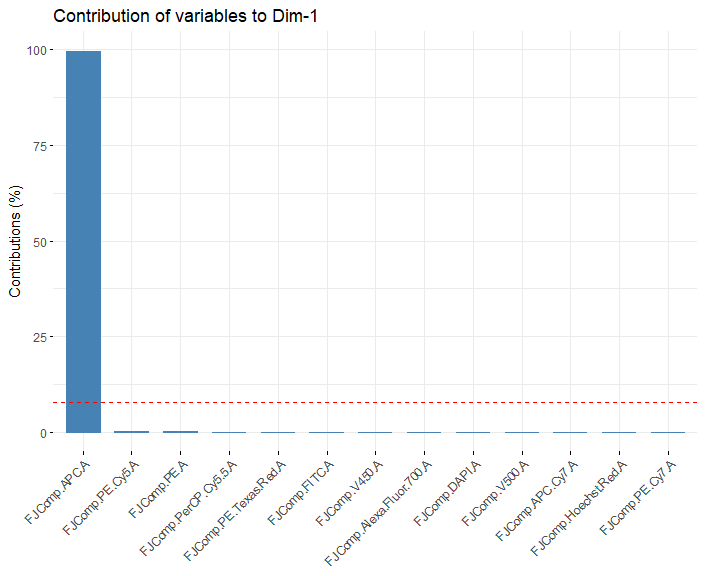
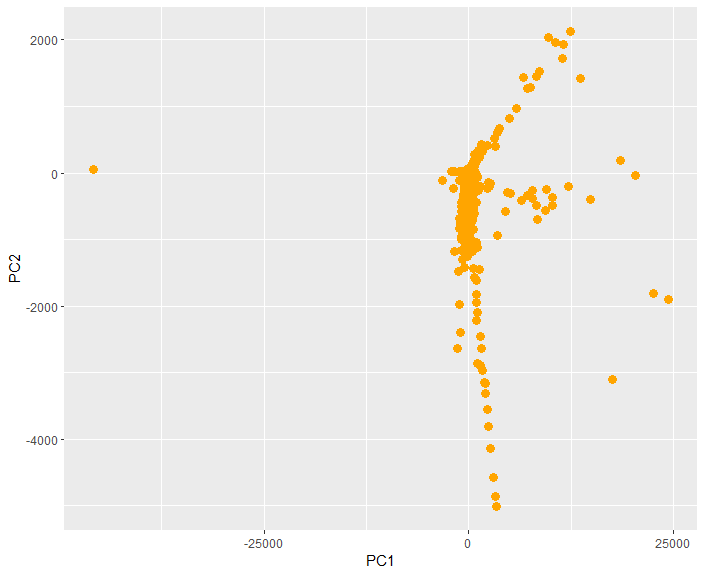


Figure 4: graph of explained variances per axis (left) and UMAP representation of data for only 2 PCA dimensions.

What could cause this issue? To have a better understanding of what is happening, we plot the contribution of each marker in the PCA. The following graph indicates that the APC marker, that is linked to the PDC cell population, contributes a lot. The plot of the two firsts PCA axis also shows that only a few points are separated by the first axis (91% of variance). In fact, PDC cells are a rare type of immune cells that secrete a lot of one specific protein. So, that is why the UMAP was of such a shape

What did we adjust ?

* **Negative expression**: we observe in the matrix expression that some values were negative. Yet negative fluorescent expression is not biologically acceptable. So, we set all negative values to 0.
* **Normalization**: we used a TMM (trimmed mean of M-values, M-values are the log fold change between each sample and a reference) from *EdgeR* package. It actually doesn’t normalize but instead calculates normalization factors. It trims off the most highly variable genes and then calculates a normalization factor that is then used to adjust the logCPM values. Here we are applying the logCPM on our whole data low and high value. When applying TMM, the value related to the APC marker expression bring a high normalisation factor in its favor and disturb the PCA. So, we remove the TMM, keep the logCPM normalization and choose to scale within the PCA, with the argument SCALE = TRUE.

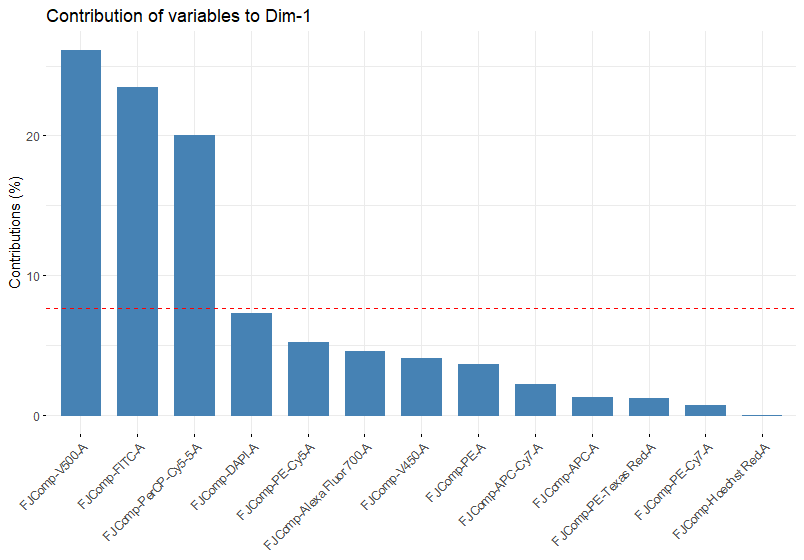
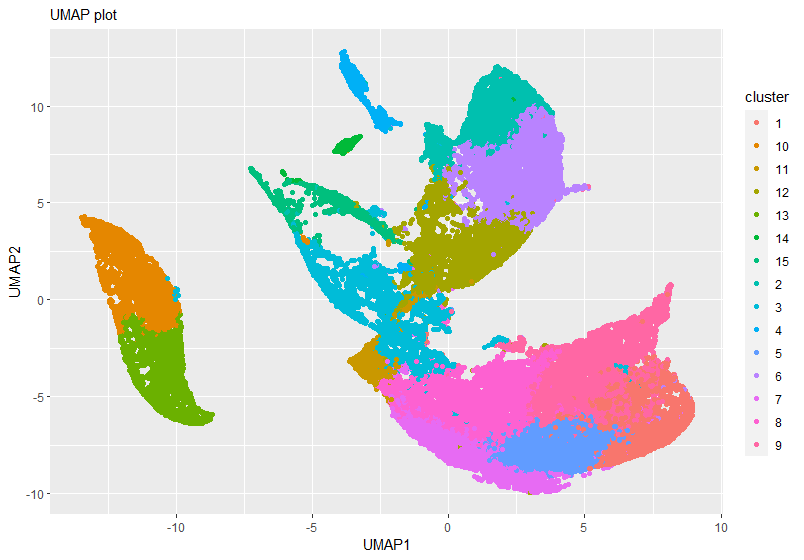
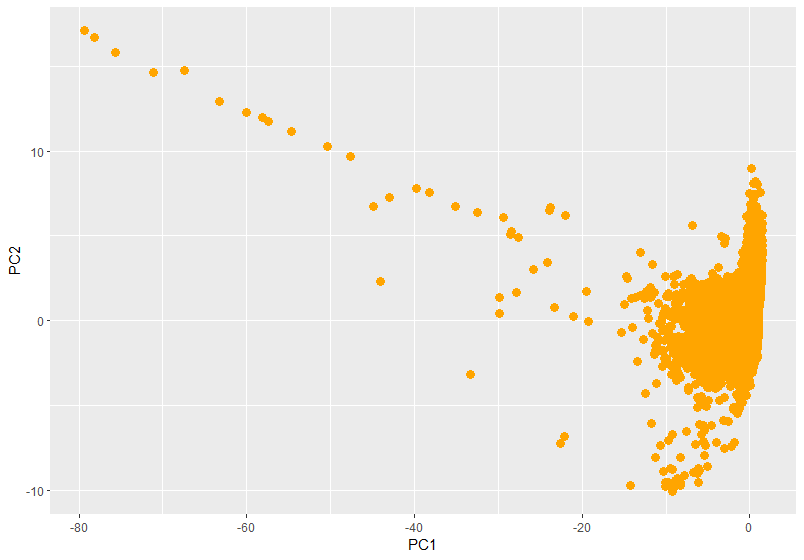


Figure 5: (top left) Graph of contribution of each marker to the PCA's dimension 1. (Top right) Plot of the first 2 PCA dimensions. (Bottom) UMAP representation of SNN+louvain clustering with 5 PCA axis

With those adjustment, we find better and more coherent results.

# Differential Expression

To identify the cell types based on our clustering, we need to identify the markers that had a significant differential expression among the clusters.

To do that, we explored the existing packages and the existing statistical tests that we could apply to our data. We only found two packages achieving this goal: *diffcyt* and *cytoTree*, both being flow cytometry data analysis pipelines containing preprocessing, clustering, and differential expression methods. However, we realized that *diffcyt* uses a FlowSOM clustering object for its differential expression method, and since we had decided not to use FlowSOM we were not able to use *diffcyt*. As for *cytoTree,* we realized that it performed differential expression analysis on the branches of the tree created by the pipeline but not on UMAP clusters. We therefore decided to drop *CytoTree* and to perform our own differential analysis method.

For that, we decided to perform a Wilcoxon test, which is a non-parametric Student test to compare two samples. We then identified top expressed markers for each cluster based on adjusted p-values and log fold changes, as is usually done to identify DEGs for RNA-Seq data. As many fold change values were negative, they were removed in the process.

# Biological Interpretation

The goal is now to interpret biologically the results of the clustering. For that we will only look at the data cd45\_pos2\_control. The statistical results show that some clusters may not have a particular set of markers with which we can identify the cellular type. Even if it is normal, it should at least connect with a big cellular type.

A cause would be that we have “over-clusterised” our dataset. To address this issue, we can modify the clustering’s resolution. We will follow a particular process and be able to interpret some clusters with certainty. First, we reduce the clustering resolution to find a few clusters and so to identify only big population types like T or B cells. Then we narrow the analysis by increasing the resolution. More clusters will form. We will be able to identify smaller and more specific cellular types that only correspond to one specific marker (like PDC for siglecX or neutrophils for Y6G). This process is possible because we only have 13 markers.

Another way to overcome “overclustering” would be to merge some clusters that present important similarities/homogeneities (same markers, in different amounts). We could apply hierarchical clustering or do that by hand.

# Discussion

* **Adaptation of the pipeline for scRNA-Seq data**

In order to achieve the goal of a joint pipeline for the analysis of flow cytometry and scRNA-Seq data, and to compare the results obtained for both types of data, we attempted to run our pipeline on scRNA-Seq data. For this, we performed pre-processing of the data in Seurat, and extracted the expression matrix as a data frame after QC and scaling. We wanted to input this data in the PCA function of the pipeline, but unfortunately the size of the data frame made the execution time of the PCA too long. So, we had to abandon this axis of study due to lack of time. However, we think that this can be a way to improve the pipeline in the future.

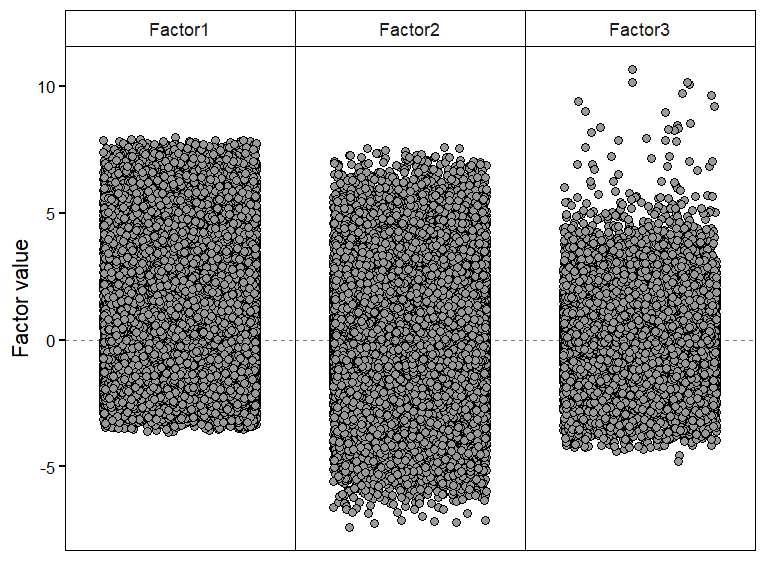
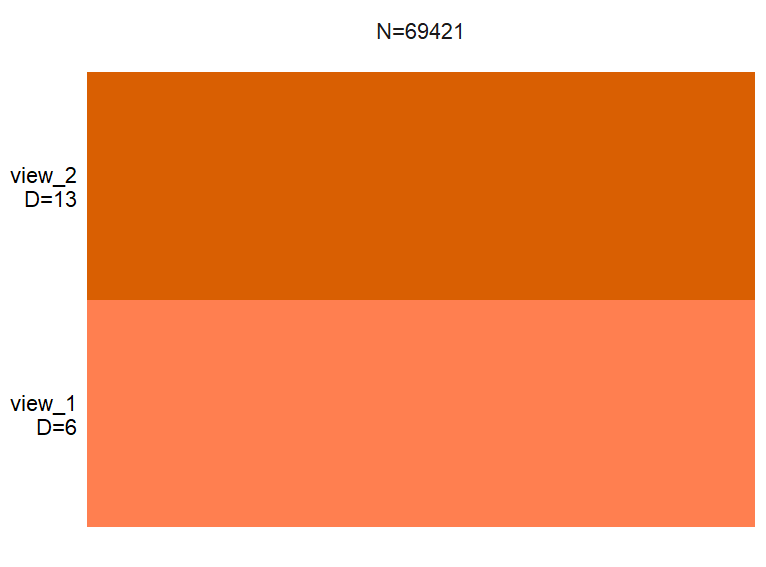
* **Integration of dimensional data**

The FSC and SSC columns of the flow cytometry expression array represent a different type of data than the data from fluorescence markers. It was therefore not possible to integrate them into the pipeline with the same format. We therefore researched multi-omics clustering techniques to integrate them. In particular, we identified the COCA (and MOFA2 packages as potential multi-omics clustering methods. COCA is a solution to integrate multi-omics data with hierarchical clustering. Unfortunately, once again, the size of the data frame did not allow us to use the COCA package because of memory size issues.

We then tried the MOFA2 package, which allows the integration of multi-omics data by training a model in order to identify the main axes of variation. To use this package, we tried using markers as features and cells as samples to create a MOFA object. We then considered the FSC/SSC data and the markers as two Views.

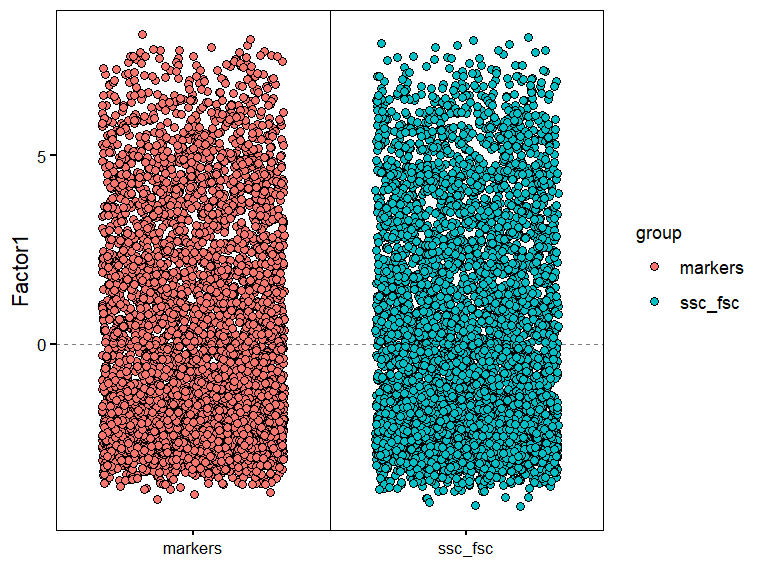
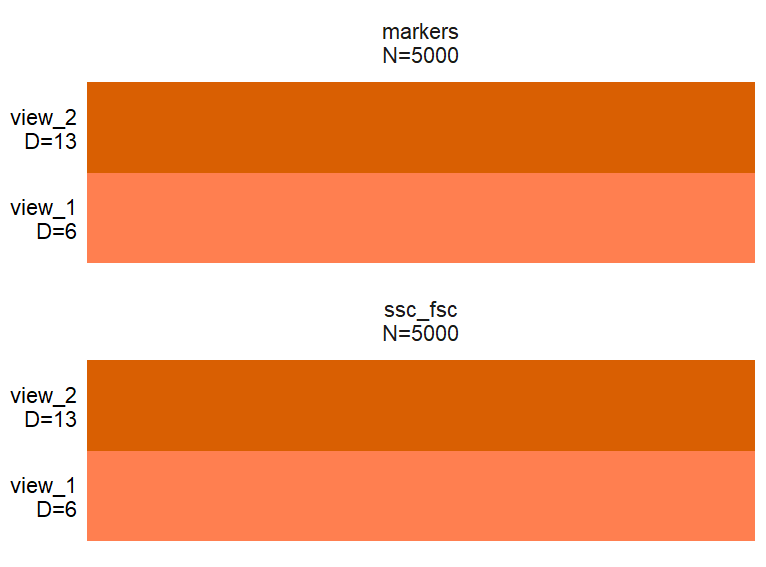
We obtained the following two graphs, however, with so many cells it is very difficult to interpret the results.

Figure 6: (left) representation of the MOFA object. (Right) Representation of the factors of the model



We then tried to represent SSC/FSC and markers as two groups, but we are not sure how to interpret the data in this case. Indeed, we did not fully understand the difference between Views and groups. We tested this with 5000 cells since it was not possible to do so with all cells, however, again the results are difficult to exploit.

Figure 7: (left) Representation of the MOFA object with groups. (Right) Representation of factor 1 of the model



We are therefore not sure that MOFA is usable in the case we are interested in. Indeed, MOFA is usually used on many more features, and on samples and not cells. We did not have the time to investigate this method further and to interpret the results. However, it may be interesting to continue testing codes using MOFA, or to test other multi-omics clustering methods.

## Conclusion

Bibliographie:

<https://support.bioconductor.org/p/9145189/>