scRNA-seq: Data Analysis

Dina Khasanova and Aleksandra Olshanova

May 22, 2023

For this homework, we will be analyzing the a dataset of 500 Human peripheral blood mononuclear cells (PBMCs) of a healthy female donor aged 25-30 freely available from 10X Genomics.

Libraries were generated from ~1675 cells (~500 cells recovered) as described in the Chromium Next GEM Single Cell 3' LT Reagent Kits (v3.1 Dual Index) User Guide (CG000399 RevB) using the Chromium Controller and sequenced on an Illumina NovaSeq 6000 to a read depth of 50k mean reads per cell.

The raw data can be found here.

We start by importing the data using the Read10X() function, which gives us a unique molecular identifier (UMI) count matrix. This matrix represents the number of molecules detected for each gene in each cell. We then created a Seurat object using the count matrix, which serves as a container for the single-cell dataset's data and analysis results.

Our data contains 703 cells (illustrated below).

```
# Load the dataset
pbmc.data <- Read10X(data.dir = "/Users/sashaolshanova/Git/machine learning course/filtered_feature_bc_m
# Initialize the Seurat object with the raw (non-normalized data).
pbmc <- CreateSeuratObject(counts = pbmc.data,  project = "pbmc3k", min.cells = 3, min.features = 200)
dim(pbmc)
```

```
## [1] 15893   703
```

Standard pre-processing workflow for scRNA-seq data in Seurat

In this work we will focus on selecting and filtering cells based on quality control (QC) metrics, normalizing and scaling the data, and identifying highly variable features.

QC and selecting cells for further analysis

Firs of all, we will explore mitochondrial contamination, as low-quality / dying cells often exhibit extensive mitochondrial contamination. We calculated mitochondrial QC metrics with the `PercentageFeatureSet()` function. This function calculates the percentage of counts originating from a set of features by using the set of all genes starting with `MT-`.

```
pbmc[["percent.mt"]] <- PercentageFeatureSet(pbmc, pattern = "^MT-")
head(pbmc[["percent.mt"]], 3)
```

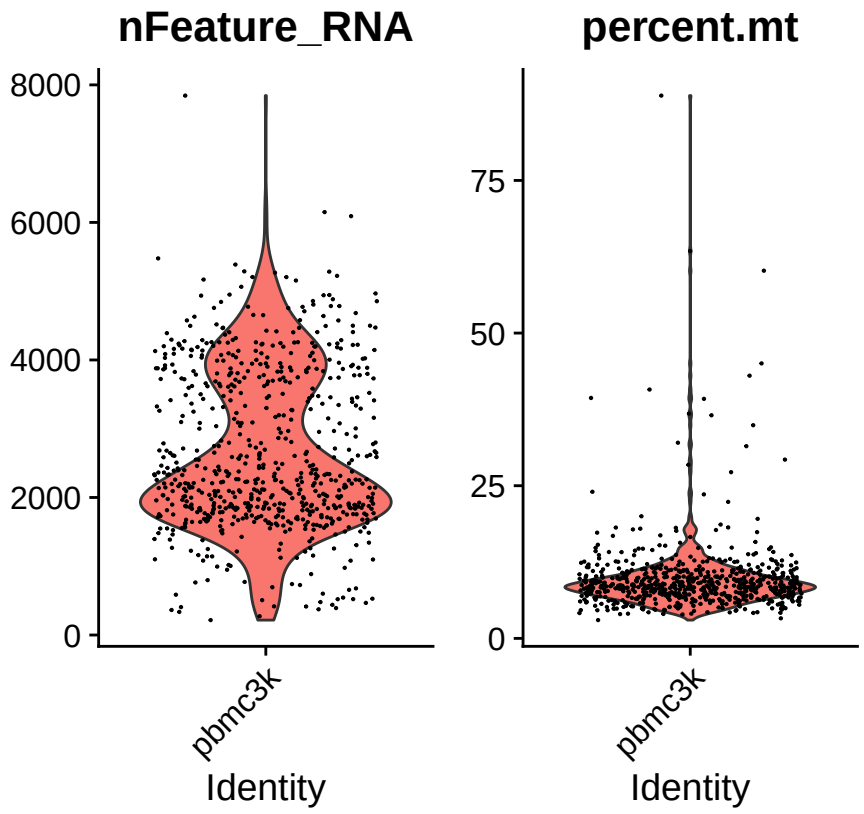
```
##                     percent.mt
## AATCACGAGGATCACG-1   6.806726
## AATCACGAGTGGCCTC-1   7.450444
## AATCACGCACAGAGAC-1   6.954040
```

After calculating the quality control (QC) metrics for mitochondria, it became crucial for us to gain insights into the distribution of these values.

Another important QC metric includes the number of unique genes detected in each cell: 1) Low-quality cells or empty droplets will often have very few genes 2) Cell doublets or multiplets may exhibit an aberrantly high gene count

To analyze this, we employed a VlnPlot approach, which is presented below.

```
#Visualize QC metrics as a VlnPlot
VlnPlot(pbmc, features = c("nFeature_RNA", "percent.mt"), ncol = 3)
```



The results clearly indicated the need to eliminate samples exhibiting mitochondrial contamination exceeding 25%. Also, we filtered cells that have unique feature counts over 5,900.

```
pbmc <- subset(pbmc, subset = nFeature_RNA < 5900 & percent.mt < 25)
dim(pbmc)
```

```
## [1] 15893   684
```

Eventually, after removing unwanted cells from the dataset, we have 684 cells left.

Normalizing the data

Once the undesirable cells have been eliminated from the dataset, the subsequent procedure involves data normalization. Our default approach is to utilize a global-scaling normalization method known as "LogNormalize." This method normalizes the feature expression measurements for each cell by considering the total expression. It then applies a scaling factor (typically set at 10,000 by default) and performs a logarithmic transformation on the outcome. The resulting normalized values can be found in the `pbmc[["RNA"]]@data`.

Identification of highly variable features (feature selection)

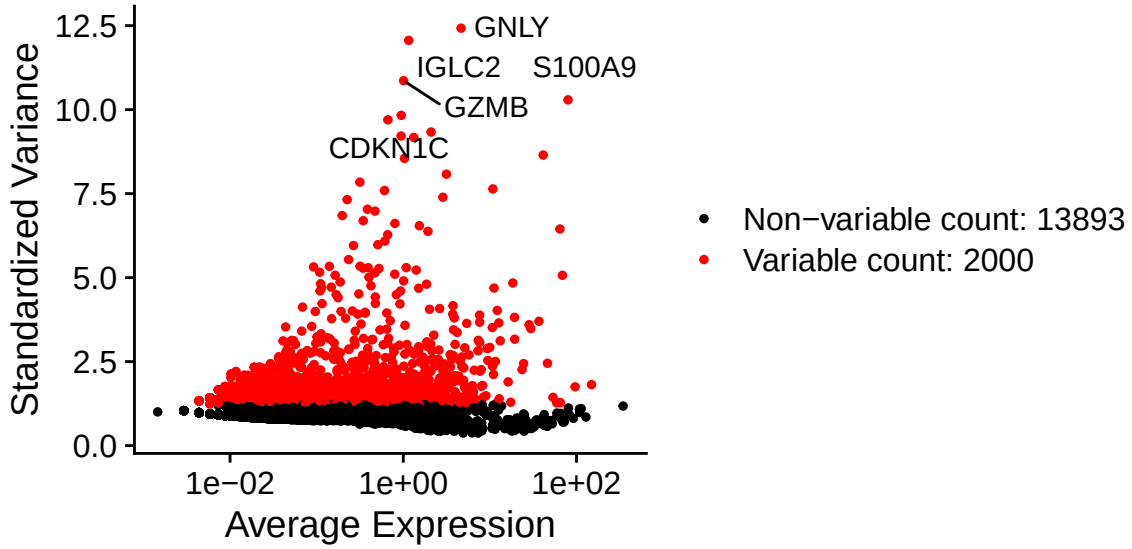
In the subsequent step, we determined a subset of features that demonstrate significant cell-to-cell variation within the dataset. These particular features display a high expression level in certain cells while being expressed at a low level in others. By concentrating on these genes during downstream analysis, we enhance the visibility of biological signals within single-cell datasets.

By default, we provided 2,000 features per dataset for further analysis, such as principal component analysis (PCA). These selected features serve as valuable inputs in downstream analytical procedures.

```
pbmc <- FindVariableFeatures(pbmc, selection.method = 'vst', nfeatures = 2000)
# Identify the 5 most highly variable genes
top5 <- head(VariableFeatures(pbmc), 5)
# plot variable features with
plot1 <- VariableFeaturePlot(pbmc)
plot2 <- LabelPoints(plot = plot1, points = top5, repel = TRUE)
```

```
## When using repel, set xnudge and ynudge to 0 for optimal results
```

```
plot2
```




```
top5
```

```
## [1] "GNLY"   "IGLC2"  "GZMB"   "S100A9" "CDKN1C"
```

Summary of the highly expressed features:

GNLY (Granulysin). Diseases associated with GNLY include Erythema Multiforme and Kyphoscoliotic Heart Disease.

IGLC2 (Immunoglobulin Lambda Constant 2). Among its related pathways are Network map of SARS-CoV-2 signaling pathway.

GZMB (Granzyme B).Diseases associated with GZMB include Peripheral T-Cell Lymphoma and Severe Cutaneous Adverse Reaction.

S100A9 (S100 Calcium Binding Protein A9). Diseases associated with S100A9 include Cystic Fibrosis and Juvenile Rheumatoid Arthritis.

CDKN1C (Cyclin Dependent Kinase Inhibitor 1C). Diseases associated with CDKN1C include Beckwith-Wiedemann Syndrome and Intrauterine Growth Retardation, Metaphyseal Dysplasia, Adrenal Hypoplasia Congenita, And Genital Anomalies.

Scaling the data

Next, we apply a linear transformation ('scaling') that is a standard pre-processing step prior to dimensional reduction techniques like PCA. The results of this are stored in `pbmc[["RNA"]]@scale.data`

Perform linear dimensional reduction

Next, we proceed with performing Principal Component Analysis (PCA) on the scaled data. By default, only the variable features that were determined earlier are utilized as input.

```
pbmc <- RunPCA(pbmc, features = VariableFeatures(object = pbmc))
```

```
## PC_ 1
## Positive:  FGL2, CYBB, MNDA, FCN1, CTSS, LYZ, PSAP, IFI30, NCF2, CST3
##     TYMP, MPEG1, CSTA, TMEM176B, IGSF6, SPI1, S100A9, SERPINA1, KCTD12, TYROBP
##     CD36, AC020656.1, AIF1, TNFAIP2, TNFSF13B, MS4A6A, CD68, FOS, GRN, FTL
## Negative:  LTB, IL32, CD3D, CD3E, BCL11B, CD3G, TRBC2, TRAC, IL7R, ARL4C
##     FCMR, TCF7, CD2, LIME1, PRKCQ-AS1, LINC00861, CD7, CCR7, CD27, GZMM
##     LEF1, TRBC1, BCL2, SYNE2, CD247, ISG20, NOSIP, IKZF3, OXNAD1, MAL
## PC_ 2
## Positive:  CLEC4C, MZB1, LILRA4, TCF4, SERPINF1, MAP1A, SPIB, DERL3, SCN9A, LINC00996
##     PACSIN1, BCL11A, DNASE1L3, EPHB1, SMPD3, LRRC26, NIBAN3, CCDC50, PLD4, JCHAIN
##     SCT, IRF8, TPM2, TSPAN13, ITM2C, PLEKHD1, IL3RA, BLNK, COBLL1, LAMP5
## Negative:  IL32, CD3D, FYB1, CD3G, CD3E, BCL11B, IL7R, TRAC, TMSB4X, ANXA1
##     LINC00861, GZMM, CD2, PRKCQ-AS1, CD247, S100A10, CD7, MT2A, S100A4, IFITM2
##     TCF7, TRBC1, ITGB2, CTSW, LEF1, OXNAD1, NEAT1, TMSB10, TXK, TRBC2
## PC_ 3
## Positive:  MS4A1, CD79A, CD79B, TNFRSF13C, LINC00926, BANK1, RALGPS2, IGHD, CD22, IGHM
##     LINC02397, FCER2, PAX5, FCRL1, VPREB3, SWAP70, AFF3, ADAM28, FCRLA, HLA-DQA1
##     HLA-DOB, CD72, P2RX5, BLK, IGKC, TCL1A, HLA-DRA, PKIG, COL19A1, POU2AF1
## Negative:  GZMB, CLIC3, C12orf75, NKG7, GZMA, CST7, SMPD3, CLEC4C, CTSW, PRF1
##     MAP1A, GNLY, SCN9A, LILRA4, PACSIN1, KLRD1, LRRC26, GZMM, FGFBP2, EPHB1
##     LIME1, GZMH, CTSC, SCT, ADGRG1, PLEKHD1, TPM2, CCL5, CCL4, HOPX
## PC_ 4
## Positive:  GNLY, NKG7, FGFBP2, PRF1, CCL4, KLRD1, KLRF1, CST7, ADGRG1, FCGR3A
```

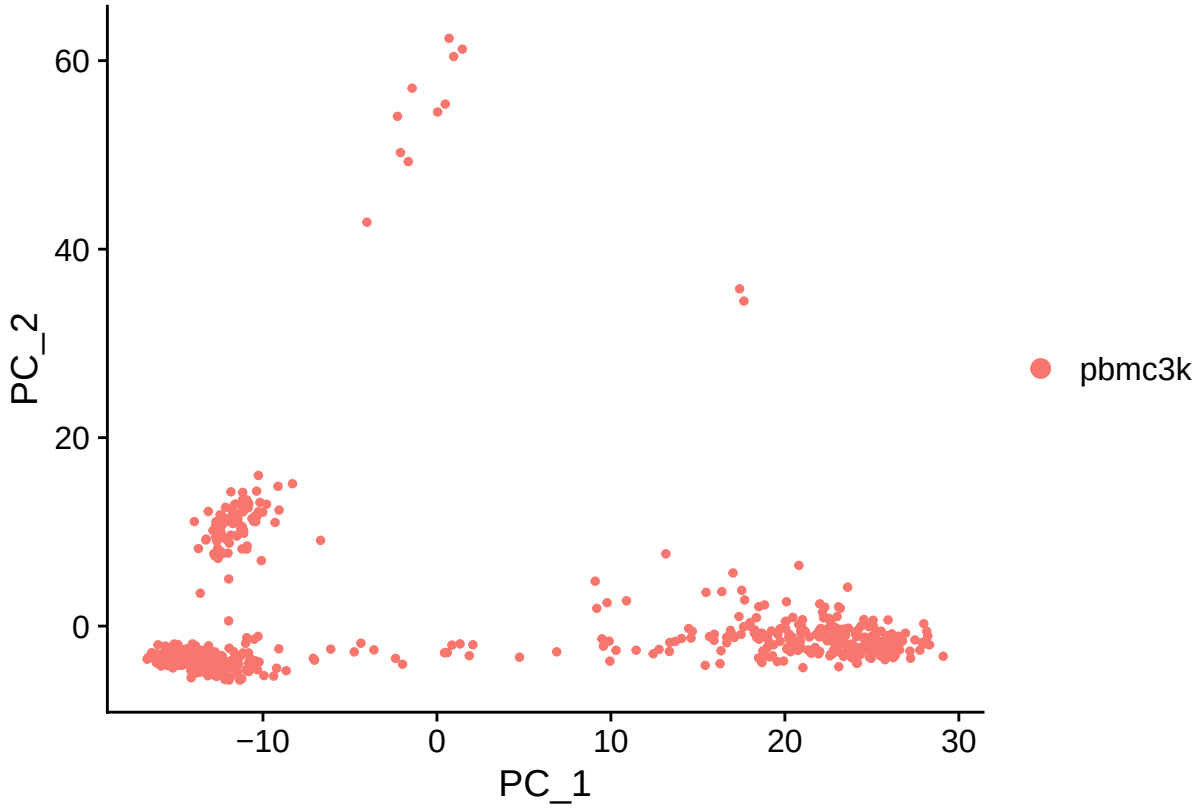
```
##     GZMA, SPON2, HOPX, GZMH, IL2RB, TBX21, CX3CR1, XCL2, ABI3, CCL5
##     CD160, PRSS23, S1PR5, CTSW, CLIC3, CHST2, PTGDR, SH2D1B, MATK, FCRL6
## Negative:  IL7R, LEF1, TCF7, MAL, NOSIP, TRABD2A, BCL11B, OXNAD1, CCR7, TRAC
##     FHIT, CD3D, VIM, CD3E, CAMK4, CD27, PRKCQ-AS1, RFLNB, PADI4, CD3G
##     SLC40A1, NUCB2, FYB1, TRAT1, LRRN3, LINC01550, LTB, CHRM3-AS2, APP, NELL2
## PC_ 5
## Positive:  S100A12, PADI4, SLC2A3, ITGAM, CYP1B1, CES1, MCEMP1, ALOX5AP, RAB27A, F5
##     THBS1, VCAN, MARC1, CR1, QPCT, METTL9, MGST1, CRISPLD2, PRF1, GNLY
##     RBP7, VNN2, S1PR3, CCL4, CTSD, DYSF, FGFBP2, JUN, BST1, S100A8
## Negative:  CDKN1C, TCF7L2, HES4, CTSL, SIGLEC10, BATF3, NAP1L1, MS4A7, CAMK1, IFITM3
##     CSF1R, RRAS, MS4A4A, LINC02345, MTSS1, NAAA, CASP5, VAMP5, CALML4, FCGR3A
##     CKB, NEURL1, SMIM25, SASH1, CXCL10, WARS, HMOX1, DUSP5, HCAR3, CALHM6
```

PCA Visualization

```
print(pbmc[['pca']], dims = 1:5, nfeatures = 5)
```

```
## PC_ 1
## Positive:  FGL2, CYBB, MNDA, FCN1, CTSS
## Negative:  LTB, IL32, CD3D, CD3E, BCL11B
## PC_ 2
## Positive:  CLEC4C, MZB1, LILRA4, TCF4, SERPINF1
## Negative:  IL32, CD3D, FYB1, CD3G, CD3E
## PC_ 3
## Positive:  MS4A1, CD79A, CD79B, TNFRSF13C, LINC00926
## Negative:  GZMB, CLIC3, C12orf75, NKG7, GZMA
## PC_ 4
## Positive:  GNLY, NKG7, FGFBP2, PRF1, CCL4
## Negative:  IL7R, LEF1, TCF7, MAL, NOSIP
## PC_ 5
## Positive:  S100A12, PADI4, SLC2A3, ITGAM, CYP1B1
## Negative:  CDKN1C, TCF7L2, HES4, CTSL, SIGLEC10
```

```
DimPlot(pbmc, reduction = 'pca')
```



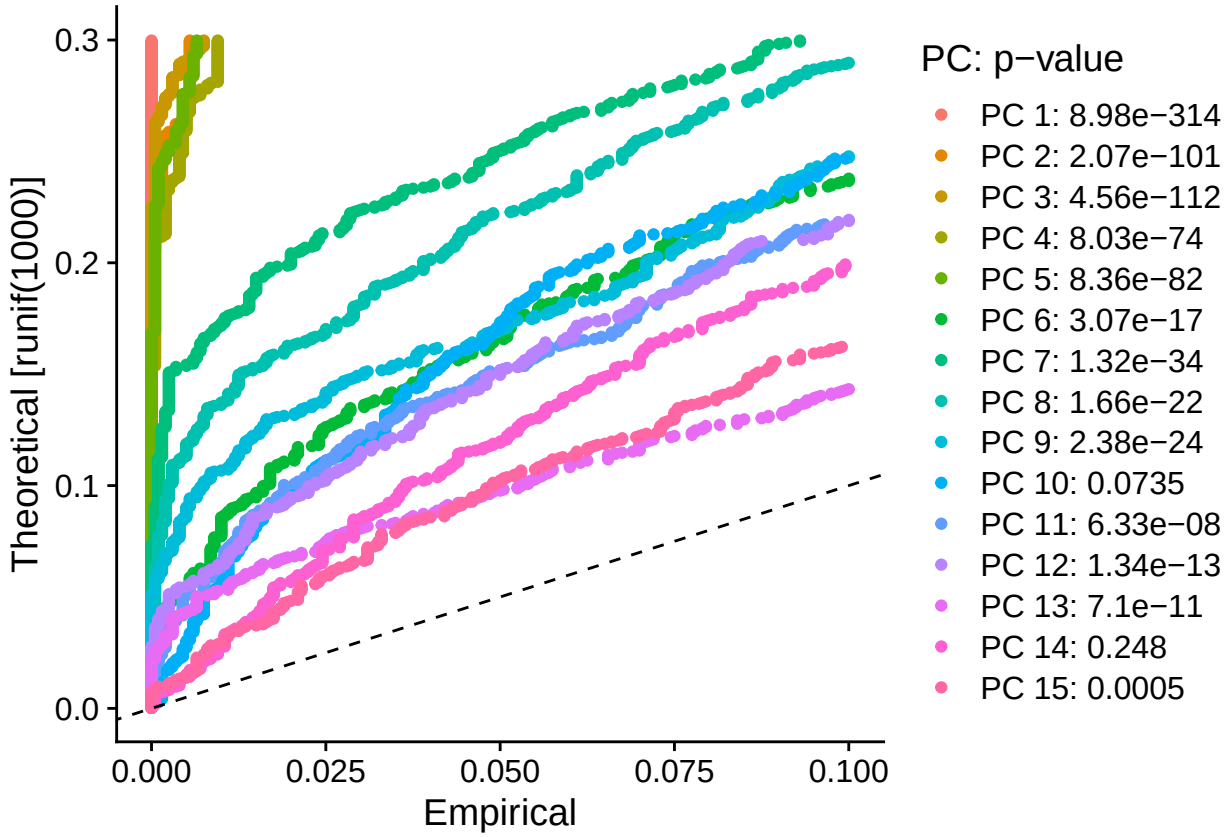
Upon examining the PCA plot, it becomes evident that the first principal component is capable of effectively separating our data points into distinct groups. In contrast, the second principal component lacks the ability to differentiate points that fall below the 20 mark on the PC-2 axis.

Determine the 'dimensionality' of the dataset

To overcome the extensive technical noise in any single feature for scRNA-seq data, Seurat clusters cells based on their PCA scores, with each PC essentially representing a 'metafeature' that combines information across a correlated feature set.

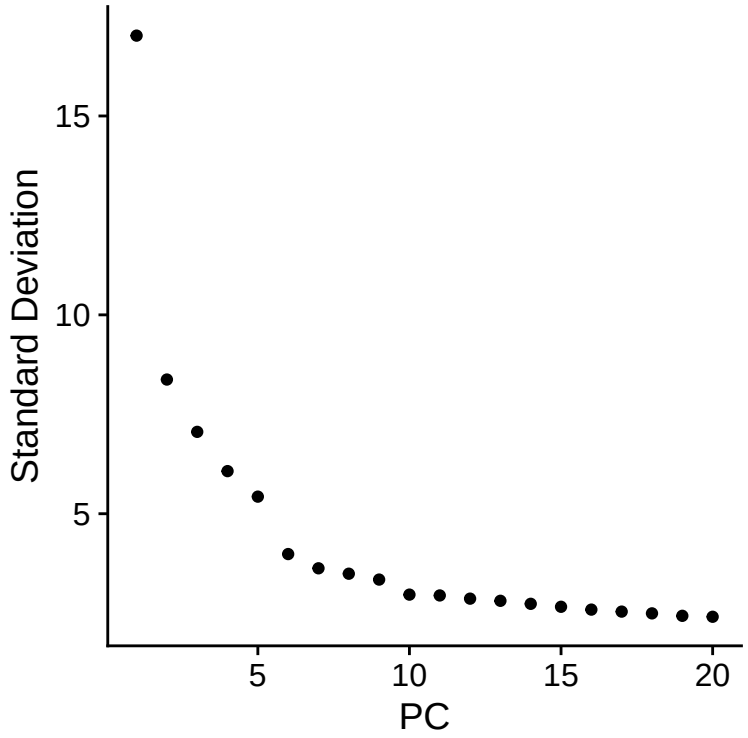
How many components should we choose to include?

```
pbmc <- JackStraw(pbmc, num.replicate = 100)
pbmc <- ScoreJackStraw(pbmc, dims = 1:20)
JackStrawPlot(pbmc, dims = 1:15)
```



The JackStrawPlot() function visualizes the distribution of p-values for each PC compared to a uniform distribution. Significant PCs show an enrichment of features with low p-values, while a drop-off in significance is observed after the first 5 PCs.

```
ElbowPlot(pbmc)
```



The ElbowPlot() function ranks PCs based on variance explained. In this case, an elbow around PC5-6 suggests that the majority of true signal is captured within the first 5 PCs.

We will choose 5.

Cluster the cells by using the KNN algorithm

- A KNN graph based on the euclidean distance in PCA space using the `FindNeighbors()` function, and taking as input the previously defined dimensionality of the dataset (first 5 PCs).
- Modularity optimization techniques such as the Louvain algorithm (default) or SLM by using the `FindClusters()` function with 5 close neibours settings.

The clusters can be found using the `Idents()` function.

```
# Look at cluster IDs of the first 5 cells
head(Idents(pbmc), 5)
```

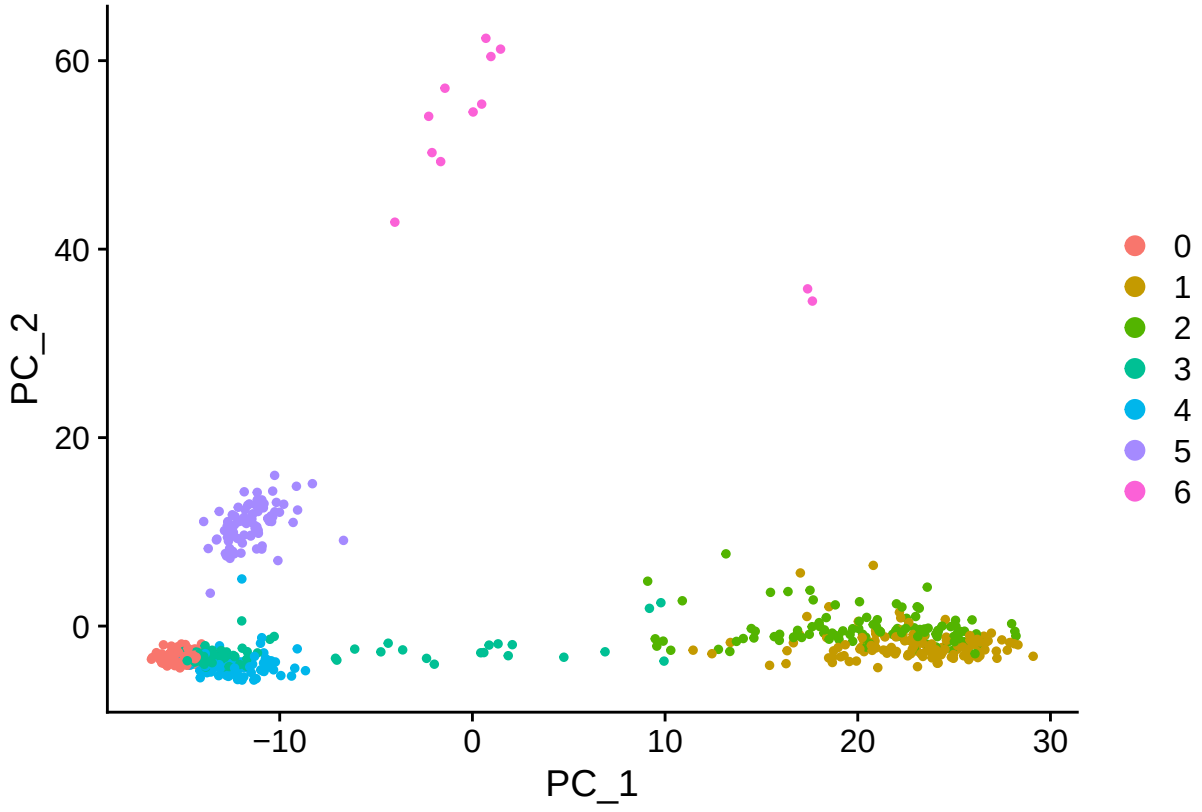
```
## AATCACGAGGATCACG-1 AATCACGAGTGGCCTC-1 AATCACGCACAGAGAC-1 AATCACGCACCAGCGT-1
##                  1                  3                  0                  2
## AATCACGCAGGTTCAT-1
##                  0
## Levels: 0 1 2 3 4 5 6
```

Overall, the KNN algorithm found 6 possible clusters. Let's visualise it based on defined clusters after implementing the KNN algorithm.

```
print(pbmc[['pca']], dims = 1:5, nfeatures = 5)
```

```
## PC_ 1
## Positive:  FGL2, CYBB, MNDA, FCN1, CTSS
## Negative:  LTB, IL32, CD3D, CD3E, BCL11B
## PC_ 2
## Positive:  CLEC4C, MZB1, LILRA4, TCF4, SERPINF1
## Negative:  IL32, CD3D, FYB1, CD3G, CD3E
## PC_ 3
## Positive:  MS4A1, CD79A, CD79B, TNFRSF13C, LINC00926
## Negative:  GZMB, CLIC3, C12orf75, NKG7, GZMA
## PC_ 4
## Positive:  GNLY, NKG7, FGFBP2, PRF1, CCL4
## Negative:  IL7R, LEF1, TCF7, MAL, NOSIP
## PC_ 5
## Positive:  S100A12, PADI4, SLC2A3, ITGAM, CYP1B1
## Negative:  CDKN1C, TCF7L2, HES4, CTSL, SIGLEC10
```

```
DimPlot(pbmc, reduction = 'pca')
```



It is evident that cluster 6 stands distinctly apart from all other groups, exhibiting a clear separation. Similarly, cluster number 5 displays a similar pattern of isolation. Regrettably, clusters 0-4 do not demonstrate a similar consistent behavior. In particular, cluster 3 appears to be distributed among clusters 0, 4, and 2, lacking a distinct pattern of its own.

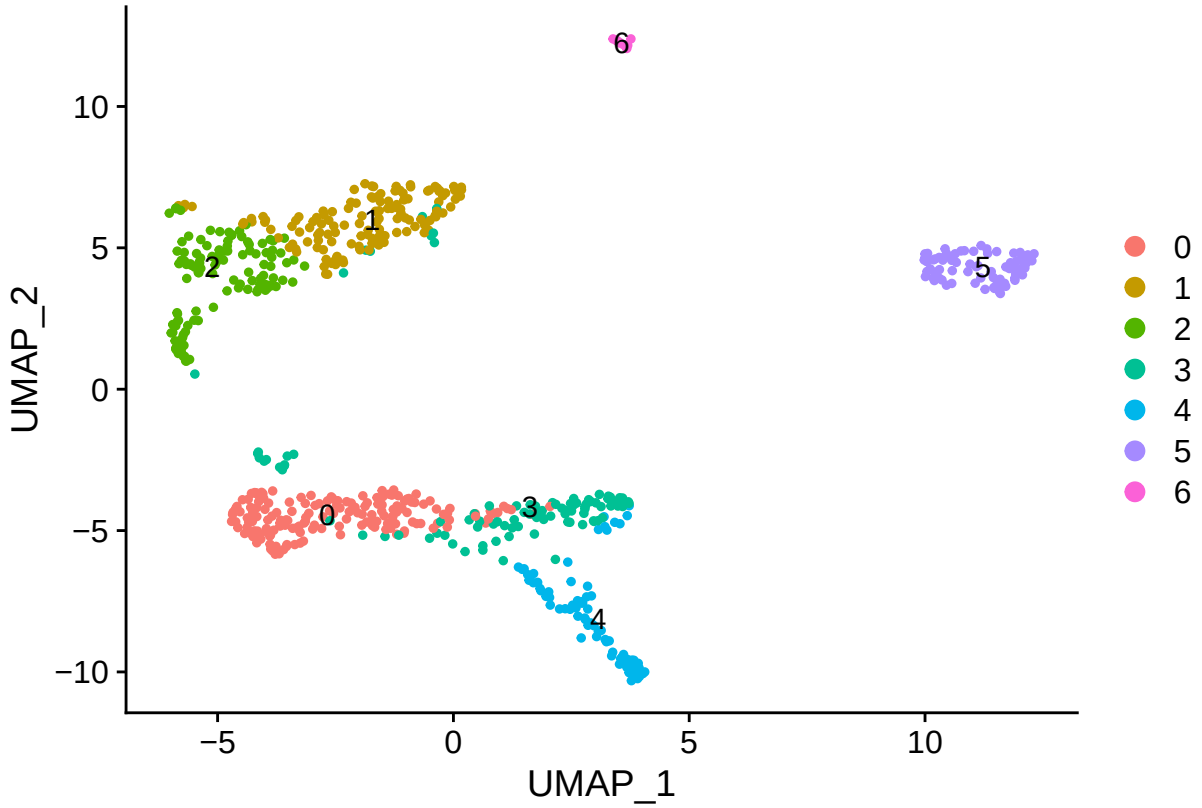
PCA focuses on preserving the global structure of the data by maximizing the variance along the principal components. It may not capture complex non-linear relationships present in the data. So, we decided to try

another dimensional reduction technique, like UMAP. UMAP often produces more visually appealing and informative visualizations.

Run non-linear dimensional reduction (UMAP)

As input to the UMAP, we used the same PCs as input to the clustering analysis.

```
umap_dimplot <- DimPlot(pbmc, reduction = 'umap', label=TRUE)
umap_dimplot
```



Upon observation, it becomes apparent that the UMAP technique enables us to effectively segregate points into distinct groups. We conducted experiments with various dimension settings and discovered that the range of 1:6 to 1:10 yields the most favorable results. However, it should be noted that increasing the number of neighbors further leads to more dispersed clusters.

Finding differentially expressed features (cluster biomarkers)

For now we would like to find markers that define clusters via differential expression. We will use `FindAllMarkers()` function, which automates an identification of positive and negative markers for all clusters. We will find markers for every cluster compared to all remaining cells and report only the positive ones with min.pct = 0.5.

```
pbmc.markers %>% group_by(cluster) %>% slice_max(n = 2, order_by = avg_log2FC)
```

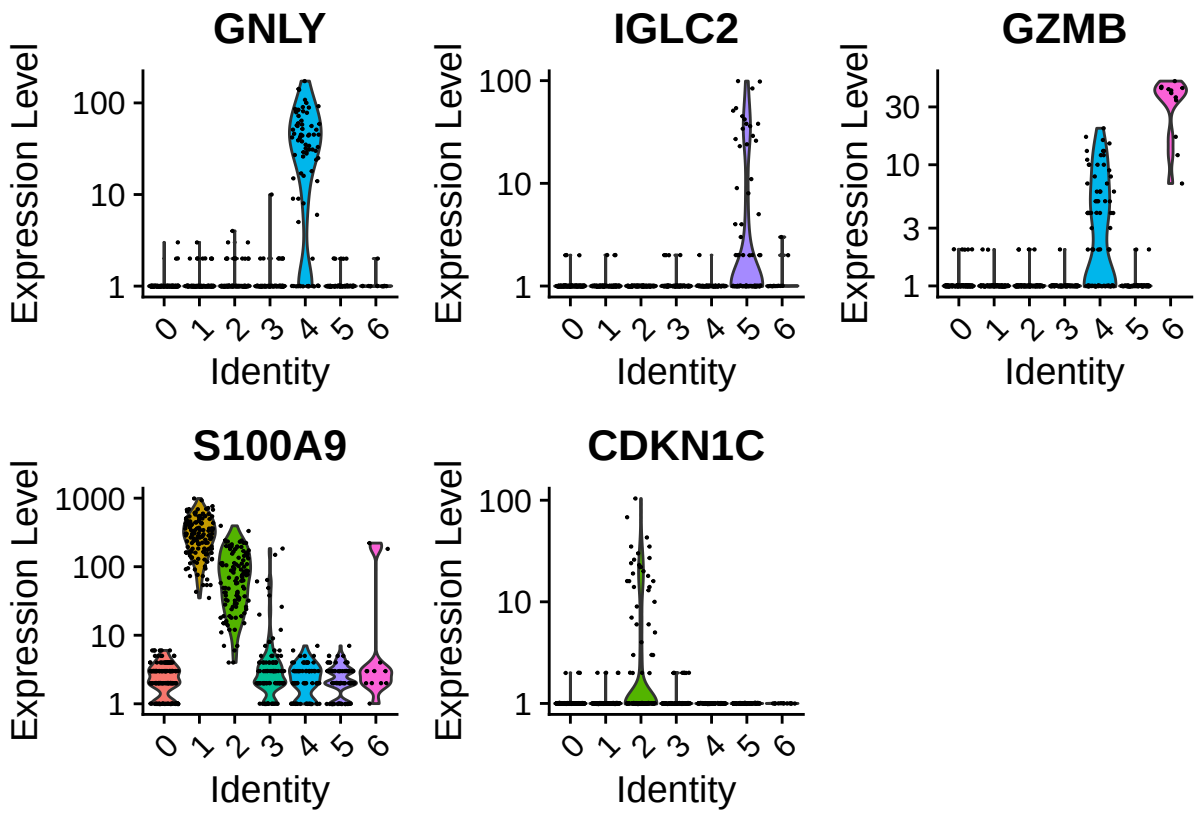
```
## # A tibble: 14 x 7
## # Groups:   cluster [7]
##       p_val avg_log2FC pct.1 pct.2 p_val_adj cluster gene
##       <dbl>      <dbl> <dbl> <dbl>     <dbl> <fct>   <chr>
##  1 4.18e-64       1.74 0.959 0.262  6.65e-60 0       BCL11B
##  2 6.00e-72       1.57 0.811 0.117  9.54e-68 0       LEF1
##  3 1.75e-69       3.75 1     0.609  2.78e-65 1       S100A8
##  4 2.30e-67       3.58 1     0.822  3.65e-63 1       S100A9
##  5 3.25e-61       2.34 0.991 0.332  5.16e-57 2       IFITM3
##  6 6.50e-55       2.26 1     0.521  1.03e-50 2       AIF1
##  7 3.21e-23       1.65 0.816 0.357  5.11e-19 3       IL7R
##  8 5.45e-15       1.36 0.857 0.677  8.66e-11 3       LTB
##  9 3.67e-74       5.94 0.797 0.066  5.83e-70 4       GNLY
## 10 1.67e-75       5.13 0.987 0.167  2.66e-71 4       NKG7
## 11 1.61e-36       5.33 0.846 0.322  2.55e-32 5       IGKC
## 12 2.82e-71       4.62 0.923 0.135  4.49e-67 5       IGHM
## 13 3.18e-16       3.92 1     0.207  5.06e-12 6       TCF4
## 14 2.58e-11       3.81 1     0.399  4.10e- 7 6       IRF8
```

The table provided may not be the most effective way to explore the distribution of different genes among the clusters. Therefore, in the following section, we will attempt to visualize the markers instead.

Visualizing marker expression.

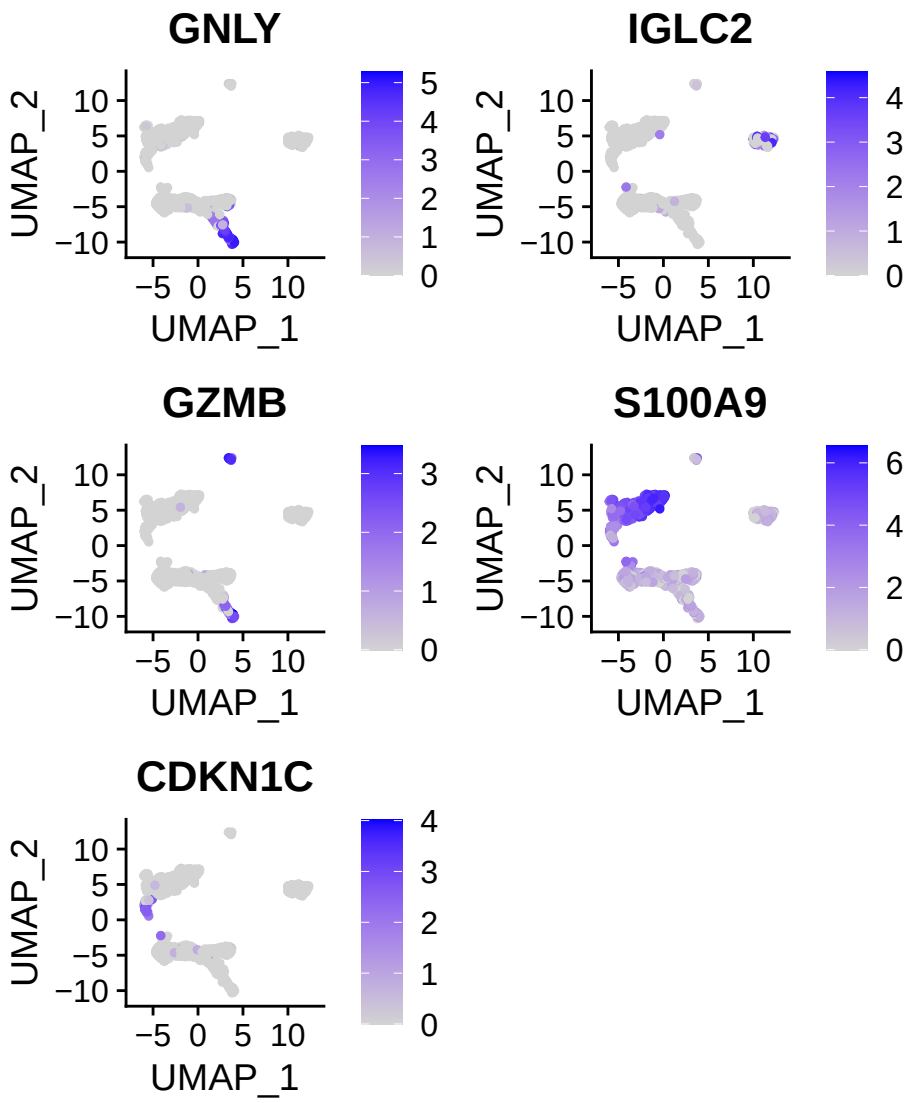
`VlnPlot()` (shows expression probability distributions across clusters), and `FeaturePlot()` (visualizes feature expression on a tSNE or PCA plot) are our most commonly used visualizations.

```
VlnPlot(pbmc, features = c("GNLY", "IGLC2", "GZMB", "S100A9", "CDKN1C"),  slot = 'counts', log = TRUE)
```




```
#  "GNLY"   "IGLC2"  "GZMB"   "S100A9" "CDKN1C"
```

```
FeaturePlot(pbmc, features = c("GNLY", "IGLC2", "GZMB", "S100A9", "CDKN1C"))
```



By employing the functions 'VlnPlot()' and 'FeaturePlot()', we can visually examine the distribution of the top 5 genes among clusters. Notably, GNLY, IGLC2, and CDKN1C distinctly represent their respective clusters: GNLY in cluster 4, IGLC2 in cluster 5, and CDKN1C in cluster 2. However, GZMB is found in two distinct clusters, specifically cluster 4 and cluster 6. Interestingly, we encounter a contradictory result regarding the gene S100A9, as it is observed across all clusters simultaneously.

Summary

In summary, our analysis of a dataset comprising 500 Human peripheral blood mononuclear cells (PBMCs) allowed us to explore various parameters successfully. Here are our key findings:

We identified the top 5 highly expressed genes, namely "GNLY," "IGLC2," "GZMB," "S100A9," and "CDKN1C." These genes are known to play significant roles in various diseases. Through the implementation of PCA and UMAP techniques, we were able to discover and visualize six distinct clusters within the dataset. Importantly, we established a connection between the top 5 identified genes and their corresponding representative clusters, highlighting the gene-cluster associations.