Expanding Acne Dataset

TamTo

2023-10-07

I’m working with a Seurat object for this analysis. Seurat objects are a representation of single-cell expression data for R, where we can analyze cell types and feature level (gene expression) of our sample conditions.

knitr::opts\_chunk$set(echo = TRUE)  
  
library(tidyverse) # load the tidyverse

## ── Attaching core tidyverse packages ──────────────────────── tidyverse 2.0.0 ──  
## ✔ dplyr 1.1.3 ✔ readr 2.1.4  
## ✔ forcats 1.0.0 ✔ stringr 1.5.0  
## ✔ ggplot2 3.4.3 ✔ tibble 3.2.1  
## ✔ lubridate 1.9.2 ✔ tidyr 1.3.0  
## ✔ purrr 1.0.2   
## ── Conflicts ────────────────────────────────────────── tidyverse\_conflicts() ──  
## ✖ dplyr::filter() masks stats::filter()  
## ✖ dplyr::lag() masks stats::lag()  
## ℹ Use the conflicted package (<http://conflicted.r-lib.org/>) to force all conflicts to become errors

library(Seurat) # we need this to work with Seurat objects

## The legacy packages maptools, rgdal, and rgeos, underpinning the sp package,  
## which was just loaded, will retire in October 2023.  
## Please refer to R-spatial evolution reports for details, especially  
## https://r-spatial.org/r/2023/05/15/evolution4.html.  
## It may be desirable to make the sf package available;  
## package maintainers should consider adding sf to Suggests:.  
## The sp package is now running under evolution status 2  
## (status 2 uses the sf package in place of rgdal)  
## Attaching SeuratObject

# Check the working directory.  
getwd()

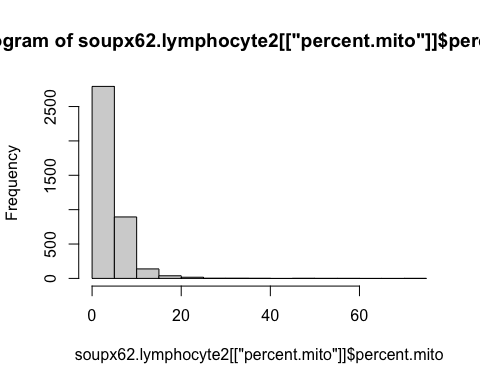
## [1] "/Users/tamto/Desktop"

# Setting the wd to where the file is located (in this case it's saved on my desktop).  
setwd("/Users/tamto/Desktop")  
  
# Time to load the data in our global environment!  
load("/Users/tamto/Desktop/soupx62.lymphocyte2.Rdata")  
  
# View the first 6 rows of the dataset.  
head(soupx62.lymphocyte2, 6)

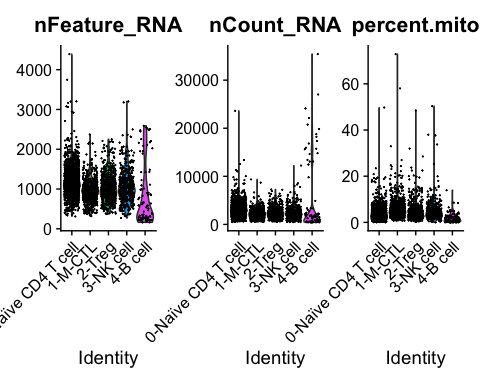
## orig.ident nCount\_RNA nFeature\_RNA percent.mito  
## L\_318A\_AAACCCAGTCAGTCCG 318 5018.969 1571 0.016923572  
## L\_318A\_AAAGAACGTGGATCAG 318 1174.791 752 0.001019867  
## L\_318A\_AAAGAACTCTATGCCC 318 3070.307 1285 0.028881840  
## L\_318A\_AAAGGTATCGCATGAT 318 2479.182 944 0.030006573  
## L\_318A\_AAAGTGATCGTAGGGA 318 3225.081 1064 0.019521774  
## L\_318A\_AACGGGAAGGTACATA 318 1931.879 845 0.028149236  
## nCount\_HTO nFeature\_HTO HTO\_maxID  
## L\_318A\_AAACCCAGTCAGTCCG 881 3 LesionalEpi  
## L\_318A\_AAAGAACGTGGATCAG 84 4 LesionalDermis  
## L\_318A\_AAAGAACTCTATGCCC 24 3 LesionalDermis  
## L\_318A\_AAAGGTATCGCATGAT 19 4 LesionalDermis  
## L\_318A\_AAAGTGATCGTAGGGA 29 2 LesionalDermis  
## L\_318A\_AACGGGAAGGTACATA 15 2 LesionalDermis  
## HTO\_secondID HTO\_margin HTO\_classification  
## L\_318A\_AAACCCAGTCAGTCCG LesionalDermis 1.2137600 LesionalDermis\_LesionalEpi  
## L\_318A\_AAAGAACGTGGATCAG NonlesionalDermis 1.9391469 LesionalDermis  
## L\_318A\_AAAGAACTCTATGCCC LesionalEpi 1.5604209 LesionalDermis  
## L\_318A\_AAAGGTATCGCATGAT NonlesionalDermis 0.8417437 LesionalDermis  
## L\_318A\_AAAGTGATCGTAGGGA NonlesionalEpi 2.0494646 LesionalDermis  
## L\_318A\_AACGGGAAGGTACATA NonlesionalEpi 1.2800965 LesionalDermis  
## HTO\_classification.global hash.ID donor  
## L\_318A\_AAACCCAGTCAGTCCG Doublet Doublet Lesional 1  
## L\_318A\_AAAGAACGTGGATCAG Singlet LesionalDermis Lesional 1  
## L\_318A\_AAAGAACTCTATGCCC Singlet LesionalDermis Lesional 1  
## L\_318A\_AAAGGTATCGCATGAT Singlet LesionalDermis Lesional 1  
## L\_318A\_AAAGTGATCGTAGGGA Singlet LesionalDermis Lesional 1  
## L\_318A\_AACGGGAAGGTACATA Singlet LesionalDermis Lesional 1  
## stim RNA\_snn\_res.0.5 seurat\_clusters  
## L\_318A\_AAACCCAGTCAGTCCG lesional 2 0  
## L\_318A\_AAAGAACGTGGATCAG lesional 2 2  
## L\_318A\_AAAGAACTCTATGCCC lesional 2 0  
## L\_318A\_AAAGGTATCGCATGAT lesional 2 1  
## L\_318A\_AAAGTGATCGTAGGGA lesional 2 0  
## L\_318A\_AACGGGAAGGTACATA lesional 2 0  
## celltype RNA\_snn\_res.0.9 RNA\_snn\_res.0.1  
## L\_318A\_AAACCCAGTCAGTCCG 0-Naïve CD4 T cell 0 0  
## L\_318A\_AAAGAACGTGGATCAG 2-Treg 4 2  
## L\_318A\_AAAGAACTCTATGCCC 0-Naïve CD4 T cell 2 0  
## L\_318A\_AAAGGTATCGCATGAT 1-M-CTL 3 1  
## L\_318A\_AAAGTGATCGTAGGGA 0-Naïve CD4 T cell 0 0  
## L\_318A\_AACGGGAAGGTACATA 0-Naïve CD4 T cell 2 0

# I like to look at it using view() since the data is so big and it's easier to visualize in Rstudio.  
view(soupx62.lymphocyte2)

# We have to be cautious of mitochondrial RNA because the mitochondria has its own RNA that can get mixed up with the RNA of our cells of interest. The PercentageFeatureSet() functions allows us to calculate the percentage of counts originating from features that contain our pattern of interest. In this case, we want to see how many genes are mitochondrial (i.e. start with "MT").  
  
soupx62.lymphocyte2[["percent.mito"]] <- PercentageFeatureSet(soupx62.lymphocyte2, pattern = "^MT-")  
  
# We can view this as a histogram to see what percentage of mitochondrial RNA the cells have. This histogram shows that most of the cells (frequency) have less than 10-15% of mitochondrial RNA. This is an acceptable cut-off.  
  
hist(soupx62.lymphocyte2[["percent.mito"]]$percent.mito)



# We can plot a violin plot to make sure we have a good reading on enough genes, there's a good reading on the gene count, and confirm low mitochondrial RNA percentage for each cell type.  
  
VlnPlot(soupx62.lymphocyte2, features = c("nFeature\_RNA", "nCount\_RNA", "percent.mito"), ncol = 3)



# These plots show that we have mostly around 1000 genes, less than 10,000 gene count, and less than 15% mitochondrial RNA content. This confirms that the dataset has been quality controlled.

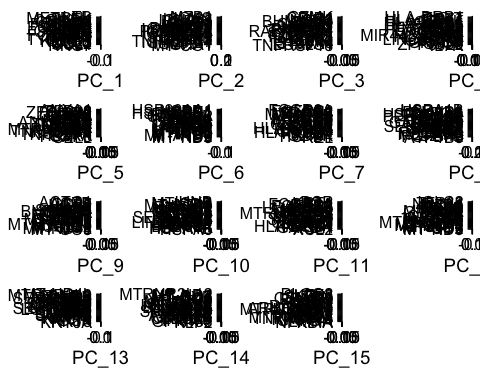
# This dataset has also already been normalized and scaled to make the values more comparable to each other. One problem is that our data is multi-dimensional (many genes) so it's not practical to analyze data with so many dimensions. Here I'm using principal component analysis (PCA) for dimensionality reduction using the RunPCA function on the Seurat object.   
  
PCALymphocyte <- RunPCA(soupx62.lymphocyte2, features = VariableFeatures(object = soupx62.lymphocyte2))

## PC\_ 1   
## Positive: LTB, CCR6, FTH1, RORA, VIM, S100A4, CD40LG, LGALS3, BATF, IL7R   
## TNFRSF4, S100A6, TMSB10, IL32, GAPDH, MAF, TSHZ2, PKM, NR3C1, ICOS   
## ZC3H12D, FRMD4B, TRBC2, AQP3, TYMP, JUNB, CTSH, CTLA4, CORO1B, SOD1   
## Negative: NKG7, CCL4, XCL2, CCL5, KLRD1, GZMB, TYROBP, CTSW, PRF1, GNLY   
## FCER1G, CST7, XCL1, GZMK, CCL4L2, GZMA, TRDC, KLRF1, CCL3, FCGR3A   
## SAMD3, GZMH, FGFBP2, MATK, CRTAM, HOPX, CCL3L1, PLEK, METRNL, IFNG   
## PC\_ 2   
## Positive: MZB1, IGKC, IGHG1, JCHAIN, DERL3, IGHG4, IGHG3, IGHG2, IGHGP, FCRL5   
## JSRP1, DNAAF1, IGHV3-23, CD79A, IGKV2-24, POU2AF1, IGKV4-1, IGHV3-33, IGF1, PRDX4   
## DPEP1, SCNN1B, IGLC2, LAMP5, SDC1, TNFRSF17, IGHV1-24, IGLV3-1, IGHA1, QPCT   
## Negative: MT-CO1, TMSB10, IL32, ANXA1, S100A4, MT-ATP6, JUNB, MT-ND3, FTH1, H3F3B   
## S100A6, ACTB, MT-ND4, MT-CYB, ZFP36L2, VIM, RGCC, MTRNR2L12, CD69, DUSP2   
## ZFP36, MT-ND2, ACTG1, LMNA, MT-ND1, MT-CO3, TRBC2, REL, PHLDA1, GAPDH   
## PC\_ 3   
## Positive: GZMK, KLF2, CCL5, GIMAP7, GIMAP4, CCL4L2, AP3M2, PASK, CCL4, PIK3R1   
## CCR7, CD8A, FCMR, SESN3, CD8B, ENC1, SELL, IFNG, MT-CO3, CCL3L1   
## GPR183, LYAR, KLRG1, MT-ND3, LINC02273, SH2D1A, DNAJB1, PDE3B, RILPL2, IGLC2   
## Negative: TNFRSF18, LGALS1, S100A4, S100A6, VIM, LMNA, CD63, CAPG, LGALS3, FCER1G   
## XCL1, SEPT11, S100A11, HOPX, RAB11FIP1, RBPJ, TYROBP, PHLDA1, KLRB1, SPINK2   
## CSF2, ACTB, GAPDH, CTSH, ENO1, BHLHE40, NCR3, PKM, CRIP1, GOLIM4   
## PC\_ 4   
## Positive: HLA-DRB1, TIGIT, CD74, HLA-DPA1, GZMK, CTLA4, HLA-DQB1, CCL5, FOXP3, HLA-DPB1   
## IL32, CCL4, TNFRSF9, HLA-DQA1, GZMH, GZMA, CD27, GBP5, CTSC, MIR4435-2HG   
## HLA-DRA, CST7, ACTB, LINC01943, CCL4L2, DUSP4, CYTOR, BATF, LAYN, PTTG1   
## Negative: IL7R, ZFP36L2, CD55, GRASP, AREG, FTH1, SPINK2, ANXA1, XCL1, FXYD7   
## KLRB1, FOS, CCR7, KIT, NFKB1, PLAC8, JUNB, FCER1G, SATB1, TYROBP   
## CD40LG, PLAUR, XBP1, BACH2, CD69, TRDC, AFF3, NFKBIA, ZFP36, CSF2   
## PC\_ 5   
## Positive: ANXA1, ZFP36, ZFP36L2, DUSP2, VIM, GZMA, LMNA, RGCC, GZMK, JUNB   
## CCL5, H3F3B, CCL4, ANKRD28, CD8A, TNFSF9, CCL4L2, CD69, CYBA, CLU   
## TNF, CD8B, DUSP1, DUSP5, GZMH, S100A4, TUBA4A, FOS, LINC01871, MT-CYB   
## Negative: SELL, FOXP3, IL2RA, TNFRSF4, CTLA4, RPS4Y1, TNFRSF18, GNLY, MTRNR2L8, GK   
## CD7, IKZF2, FCER1G, LAYN, SATB1, F5, TBC1D4, STAM, GBP5, PMAIP1   
## TYROBP, LAIR2, ENTPD1, HACD1, CARD16, IL18R1, MT-ND4L, TXK, PLAC8, BEX3

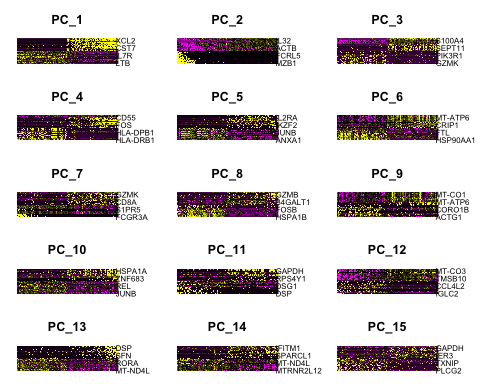
# I chose to look at 15 different dimensional variations since I'm trying to expand the dataset. We can visualize the difference in genes of each condition in each dimension in various ways (i.e. print() or VizDimLoadings()).  
  
print(PCALymphocyte[['pca']], dims = 1:15, nfeatures = 5)

## PC\_ 1   
## Positive: LTB, CCR6, FTH1, RORA, VIM   
## Negative: NKG7, CCL4, XCL2, CCL5, KLRD1   
## PC\_ 2   
## Positive: MZB1, IGKC, IGHG1, JCHAIN, DERL3   
## Negative: MT-CO1, TMSB10, IL32, ANXA1, S100A4   
## PC\_ 3   
## Positive: GZMK, KLF2, CCL5, GIMAP7, GIMAP4   
## Negative: TNFRSF18, LGALS1, S100A4, S100A6, VIM   
## PC\_ 4   
## Positive: HLA-DRB1, TIGIT, CD74, HLA-DPA1, GZMK   
## Negative: IL7R, ZFP36L2, CD55, GRASP, AREG   
## PC\_ 5   
## Positive: ANXA1, ZFP36, ZFP36L2, DUSP2, VIM   
## Negative: SELL, FOXP3, IL2RA, TNFRSF4, CTLA4   
## PC\_ 6   
## Positive: HSP90AA1, HSPD1, HSPE1, HSP90AB1, HSPH1   
## Negative: MT-ND3, MT-ND2, MT-ATP6, MT-ND1, MT-ND5   
## PC\_ 7   
## Positive: FCGR3A, FGFBP2, KLRF1, SPON2, MT-CO3   
## Negative: REL, HSPD1, GZMK, XCL1, FXYD2   
## PC\_ 8   
## Positive: HSPA1B, HSPA1A, DNAJB1, DUSP1, FOS   
## Negative: ATP1B3, FGFBP2, GZMB, METRNL, FCGR3A   
## PC\_ 9   
## Positive: ACTG1, ACTB, CD27, UGP2, IL32   
## Negative: MT-CO3, MT-ND4L, MT-CO1, MT-CO2, CCL20   
## PC\_ 10   
## Positive: JUNB, MT-CYB, ICOS, MT-ATP6, DUSP2   
## Negative: HSPA6, HSPA1B, HSPA1A, HSPB1, DNAJB4   
## PC\_ 11   
## Positive: DSP, KRT1, FCGR3A, DSC3, FGFBP2   
## Negative: XCL2, XCL1, GAPDH, TNFSF4, HLA-DRB1   
## PC\_ 12   
## Positive: IGLC2, ZFP36, NR4A1, MZB1, CCL3   
## Negative: MT-ND3, MT-CO2, MT-CO3, MT-ND4, RPS4Y1   
## PC\_ 13   
## Positive: MT-ND4L, MTRNR2L8, MT-ND3, TNFAIP3, SQSTM1   
## Negative: KRT6A, KRT5, DSP, KRT1, KRT14   
## PC\_ 14   
## Positive: MTRNR2L12, MT-ND4, MT-CO2, MT-CYB, MT-CO1   
## Negative: KLF2, GPR183, IFITM1, MAF, CLEC2B   
## PC\_ 15   
## Positive: PLCG2, IKZF3, SLFN5, TTN, CEMIP2   
## Negative: NFKBIA, LMNA, GAPDH, RPS4Y1, TNFRSF18

VizDimLoadings(PCALymphocyte, dims = 1:15, reduction = 'pca')

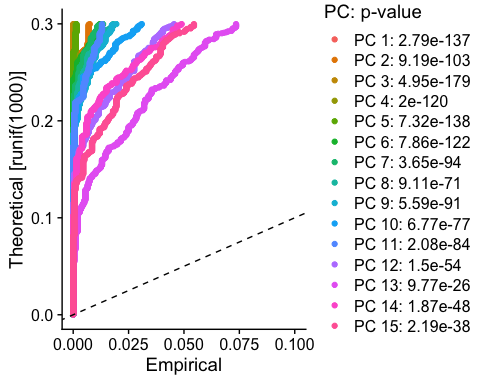


# DimHeatmap() function allows to explore heterogeneity in the dataset and allows us to decide which PCs to include for downstream analyses.  
  
DimHeatmap(PCALymphocyte, dims = 1:15, cells = 500, balanced = TRUE)



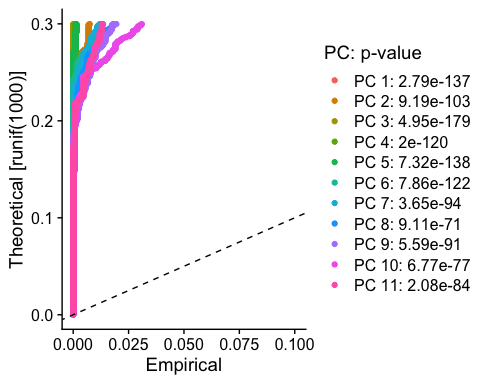
# We can create a JackStraw plot using the JackStrawPlot() function to verify which PCs are significant. JackStraw() and ScoreJackStraw() compares the distribution of p-values for each PC with a uniform distribution (dashed line in the plot). Significant PCs show a strong enrichment of features with low p-values (solid curves in the plot). The closer / more similar they are to the dashed line, the less meaningful/significant they are.  
  
PCALymphocyte <- JackStraw(PCALymphocyte, num.replicate = 100)  
PCALymphocyte <- ScoreJackStraw(PCALymphocyte, dims = 1:15)  
JackStrawPlot(PCALymphocyte, dims = 1:15)

## Warning: Removed 21000 rows containing missing values (`geom\_point()`).

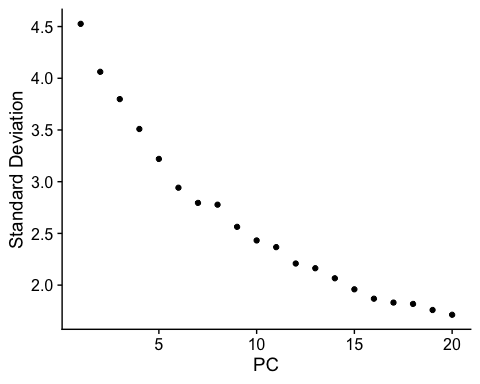


# The plot shows all 15 lines are above the dashed line, but PC 12-15 start to deviate a bit. I'm going to cut it off at PC 11.  
  
JackStrawPlot(PCALymphocyte, dims = 1:11)

## Warning: Removed 15400 rows containing missing values (`geom\_point()`).



# We can verify this another way using an Elbow plot with the ElbowPlot() function that ranks PCs based on the percentage of variance.   
  
ElbowPlot(PCALymphocyte)



# We see there is also an elbow at PC 11 so it verifies the cut off I'm choosing here.

# This is a dimensionality reduction technique used for visualization. It captures the manifold (topology/shape) of the data organization in higher dimensions and embeds a neighborhood of points. This uses the K-NN/KNN (K-nearest neighbor) algorithm to find similar gene expression profiles between 2 cells.  
  
NewLymphocyteClusters <- FindNeighbors(PCALymphocyte, dims = 1:11)

## Computing nearest neighbor graph

## Computing SNN

NewLymphocyteClusters <- FindClusters(PCALymphocyte, resolution = 0.5)

## Modularity Optimizer version 1.3.0 by Ludo Waltman and Nees Jan van Eck  
##   
## Number of nodes: 3893  
## Number of edges: 125380  
##   
## Running Louvain algorithm...  
## Maximum modularity in 10 random starts: 0.8607  
## Number of communities: 11  
## Elapsed time: 0 seconds

## Warning: Adding a command log without an assay associated with it

# Now we can verify what the cluster IDs of the first 5 cells are and that we have a total of 11 clusters since that was our cut-off for cell types.   
head(Idents(NewLymphocyteClusters), 5)

## L\_318A\_AAACCCAGTCAGTCCG L\_318A\_AAAGAACGTGGATCAG L\_318A\_AAAGAACTCTATGCCC   
## 4 3 1   
## L\_318A\_AAAGGTATCGCATGAT L\_318A\_AAAGTGATCGTAGGGA   
## 2 0   
## Levels: 0 1 2 3 4 5 6 7 8 9 10

# Looks like we have 11 total levels (cell types) and they are named by number. We can create a UMAP to view the graph-based clusters and verify these cell types cluster on the map.  
  
NewLymphocyteClusters <- RunUMAP(NewLymphocyteClusters, reduction = "pca", dims = 1:11)

## Warning: The default method for RunUMAP has changed from calling Python UMAP via reticulate to the R-native UWOT using the cosine metric  
## To use Python UMAP via reticulate, set umap.method to 'umap-learn' and metric to 'correlation'  
## This message will be shown once per session

## 16:23:11 UMAP embedding parameters a = 0.9922 b = 1.112

## 16:23:11 Read 3893 rows and found 11 numeric columns

## 16:23:11 Using Annoy for neighbor search, n\_neighbors = 30

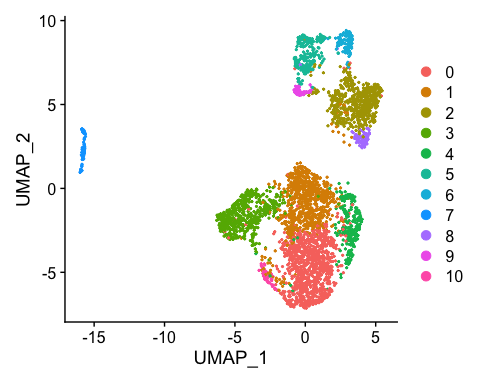
## 16:23:11 Building Annoy index with metric = cosine, n\_trees = 50

## 0% 10 20 30 40 50 60 70 80 90 100%

## [----|----|----|----|----|----|----|----|----|----|

## \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*|  
## 16:23:11 Writing NN index file to temp file /var/folders/xk/m16nzsr50mj5\_cn5n88svr\_w0000gp/T//RtmpZ4Gl5x/file58581c0c222f  
## 16:23:11 Searching Annoy index using 1 thread, search\_k = 3000  
## 16:23:12 Annoy recall = 100%  
## 16:23:12 Commencing smooth kNN distance calibration using 1 thread with target n\_neighbors = 30  
## 16:23:13 Initializing from normalized Laplacian + noise (using irlba)  
## 16:23:13 Commencing optimization for 500 epochs, with 152338 positive edges  
## 16:23:16 Optimization finished

DimPlot(NewLymphocyteClusters, reduction = "umap")



# Now we need to find out what each of these cells are. We can do this by finding out their top genes with FindAllMarkers() and deduce which cells they are based on the expression of these genes. I like to find the top 10 markers of each cluster, min.pct argument is used to restrict to features detected in a minimum fraction of the chosen percentage of cells (i.e. here it has to be expressed in at least 25% of the cells). I chose a minimum log2 fold change threshold as 0.25 for average expression of a gene in one cluster relative to other clusters (i.e. logfc.threshold = 0.25).  
  
cluster\_markers <- FindAllMarkers(NewLymphocyteClusters,  
 only.pos = TRUE,   
 min.pct = 0.25,   
 logfc.threshold = 0.25)

## Calculating cluster 0

## Calculating cluster 1

## Calculating cluster 2

## Calculating cluster 3

## Calculating cluster 4

## Calculating cluster 5

## Calculating cluster 6

## Calculating cluster 7

## Calculating cluster 8

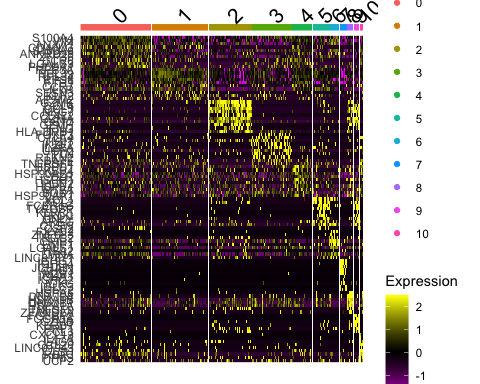
## Calculating cluster 9

## Calculating cluster 10

# You can view the top genes for each cluster using slice\_max().  
cluster\_markers %>%  
 group\_by(cluster) %>%  
 slice\_max(n = 1, order\_by = avg\_log2FC)

## # A tibble: 11 × 7  
## # Groups: cluster [11]  
## p\_val avg\_log2FC pct.1 pct.2 p\_val\_adj cluster gene   
## <dbl> <dbl> <dbl> <dbl> <dbl> <fct> <chr>   
## 1 5.42e-178 1.13 0.988 0.815 1.59e-173 0 S100A4  
## 2 4.28e- 50 1.02 0.286 0.093 1.26e- 45 1 SESN3   
## 3 1.80e-243 3.60 0.474 0.033 5.28e-239 2 CCL4L2  
## 4 5.19e- 70 1.58 0.582 0.283 1.52e- 65 3 UGP2   
## 5 9.01e- 65 1.79 0.898 0.52 2.65e- 60 4 HSPD1   
## 6 0 4.32 0.838 0.062 0 5 XCL1   
## 7 1.72e- 42 3.10 0.264 0.029 5.06e- 38 6 HBA2   
## 8 4.80e-213 11.1 0.656 0.027 1.41e-208 7 IGKC   
## 9 2.00e-119 4.01 0.602 0.04 5.88e-115 8 HSPA6   
## 10 5.20e-188 4.34 0.986 0.065 1.53e-183 9 GZMB   
## 11 4.70e- 92 4.78 0.396 0.012 1.38e- 87 10 CXCL13

# I want to make a heatmap of the top 10 genes since this gives us more information than just one top gene.  
top10markers <- cluster\_markers %>%  
 group\_by(cluster) %>%  
 top\_n(n = 10, wt = avg\_log2FC)  
  
DoHeatmap(NewLymphocyteClusters, features = top10markers$gene)



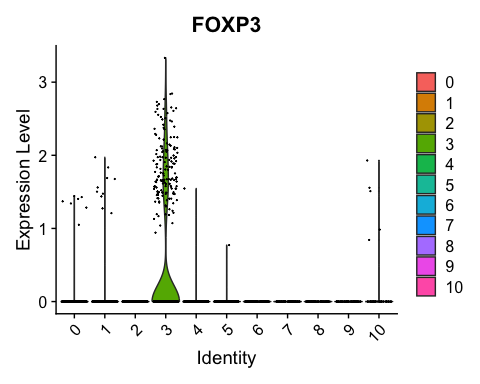
# Since the heatmap can get quite overwhelming with a lot of clusters, we can look at one cluster at a time. I'm going to start with cluster 0 and then this would be repeated for each cluster to find out their top genes.  
  
cluster0.markers <- FindMarkers(NewLymphocyteClusters, ident.1 = 0, min.pct = 0.25)  
head(cluster0.markers, n = 10)

## p\_val avg\_log2FC pct.1 pct.2 p\_val\_adj  
## S100A4 5.419047e-178 1.1316279 0.988 0.815 1.592658e-173  
## VIM 1.700104e-119 0.8895228 0.993 0.860 4.996604e-115  
## B2M 6.184956e-113 0.5123012 0.999 0.994 1.817758e-108  
## ANXA1 6.488899e-103 0.8646855 0.965 0.675 1.907087e-98  
## MYL12A 3.256195e-100 0.7691072 0.952 0.760 9.569957e-96  
## CD40LG 3.662853e-97 0.9740685 0.484 0.150 1.076512e-92  
## S100A6 5.912430e-92 0.8662376 0.966 0.757 1.737663e-87  
## CAPG 1.201263e-89 0.8324390 0.482 0.157 3.530513e-85  
## LMNA 8.818674e-84 0.8378498 0.908 0.634 2.591808e-79  
## ANKRD28 1.632434e-83 0.8420013 0.533 0.200 4.797722e-79

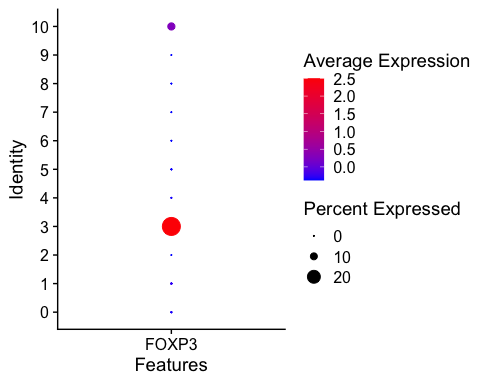
# If it is difficult to distinguish a certain cluster from another cluster, we can use the following line of code. If we want to distinguish one cluster from 2 clusters, we can change ident.2 = c() to include more than one cluster.  
  
cluster8.markers <- FindMarkers(NewLymphocyteClusters, ident.1 = 8, ident.2 = 2, min.pct = 0.25)  
head(cluster8.markers, n = 10)

## p\_val avg\_log2FC pct.1 pct.2 p\_val\_adj  
## HSPA6 4.064077e-70 4.596083 0.602 0.018 1.194432e-65  
## HSPA1B 8.304835e-44 3.303275 0.988 0.453 2.440791e-39  
## DNAJB4 2.479526e-42 2.539914 0.627 0.085 7.287327e-38  
## HSPA1A 5.533323e-40 3.565678 0.964 0.547 1.626244e-35  
## DNAJB1 1.305748e-36 2.955465 0.964 0.621 3.837593e-32  
## DNAJA4 2.572017e-36 2.018598 0.434 0.033 7.559157e-32  
## BAG3 1.880748e-34 1.872556 0.386 0.025 5.527518e-30  
## HSPB1 2.743383e-34 2.259889 0.735 0.178 8.062804e-30  
## HSP90AA1 3.260366e-31 2.237617 0.988 0.890 9.582217e-27  
## HSPH1 9.261710e-31 2.407777 0.783 0.261 2.722017e-26

# I'm more of a visual analyzer, so I recommend making violin plots or dot plots of unique genes that are known to be specific for each cell type. In this example, FOXP3 is a gene specific for T regulatory cells (Treg). Looks like cluster 3 is a Treg cell cluster. We can then repeat this with more unique genes to find out the remaining cell types for each cluster.  
  
VlnPlot(NewLymphocyteClusters, features = c("FOXP3"))



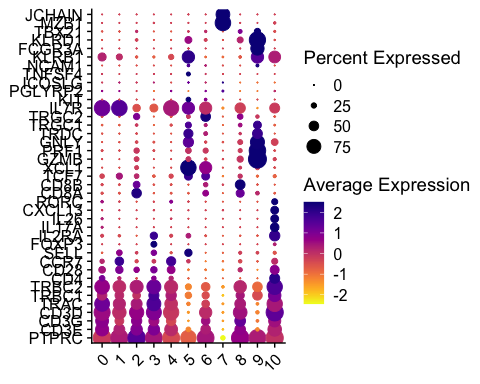
DotPlot(NewLymphocyteClusters, features = c("FOXP3"), cols = c("blue", "red"))



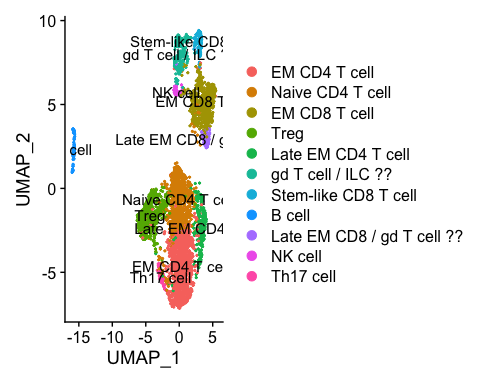
# Although we might be certain of a cluster based on specific genes, we also need to verify that these cell types are actually true by comparing the genes across all clusters. We can do this by plotting genes and clusters using the scCustomize package.   
  
# install.packages("scCustomize")  
library(scCustomize)

## scCustomize v1.1.3  
## If you find the scCustomize useful please cite.  
## See 'samuel-marsh.github.io/scCustomize/articles/FAQ.html' for citation info.

genes <- c("PTPRC", "CD3E", "CD3G", "CD3D", "TRAC", "TRBC1", "TRBC2", "CD4", "CD28", "CCR7", "SELL", "FOXP3", "IL2RA", "IL17A", "IL26", "CXCL13", "RORC", "CD8A", "CD8B", "TCF7", "XCL1", "GZMB", "PRF1", "GNLY", "TRDC", "TRGC1", "TRGC2", "IL7R", "KIT", "PGLYRP2", "ICOSLG", "TNFSF4", "NCAM1", "KLRB1", "FCGR3A", "KLRD1", "TBX21", "MZB1", "JCHAIN")  
  
DotPlot\_scCustom(seurat\_object = NewLymphocyteClusters, features = genes, flip\_axes = T, x\_lab\_rotate = T)



# Now I've found out what each cluster may be based on their specific genes. It's time to rename the clusters with their correct cell type.  
  
new.cluster.ids <- c("EM CD4 T cell", "Naive CD4 T cell", "EM CD8 T cell", "Treg", "Late EM CD4 T cell", "gd T cell / ILC ??", "Stem-like CD8 T cell", "B cell", "Late EM CD8 / gd T cell ??", "NK cell", "Th17 cell")  
  
names(new.cluster.ids) <- levels(NewLymphocyteClusters)  
  
NewLymphocyteClusters <- RenameIdents(NewLymphocyteClusters, new.cluster.ids)  
  
# View the new UMAP with the updated clusters!  
DimPlot(NewLymphocyteClusters, reduction = 'umap', label = TRUE, pt.size = 0.4)



# I want to reorder the cell types so that it's easier to analyze when we make downstream analyses. Make the new ordered cell types the active identities.  
  
NewLymphocyteClusters@active.ident <- factor(NewLymphocyteClusters@active.ident,  
 levels = c("Naive CD4 T cell",  
 "EM CD4 T cell",  
 "Late EM CD4 T cell",  
 "Treg",  
 "Th17 cell",  
 "Stem-like CD8 T cell",  
 "EM CD8 T cell",  
 "Late EM CD8 / gd T cell ??",  
 "gd T cell / ILC ??",  
 "NK cell",  
 "B cell")  
 )  
  
NewLymphocyteClusters$celltype <- Idents(NewLymphocyteClusters)  
  
# Verify each cluster and their top genes in an updated Heatmap that shows all the renamed clusters. We can also view it as a stacked dot plot again but now with named, reordered clusters.  
  
cluster\_markers <- FindAllMarkers(NewLymphocyteClusters, only.pos = TRUE, min.pct = 0.25, logfc.threshold = 0.25)

## Calculating cluster Naive CD4 T cell

## Calculating cluster EM CD4 T cell

## Calculating cluster Late EM CD4 T cell

## Calculating cluster Treg

## Calculating cluster Th17 cell

## Calculating cluster Stem-like CD8 T cell

## Calculating cluster EM CD8 T cell

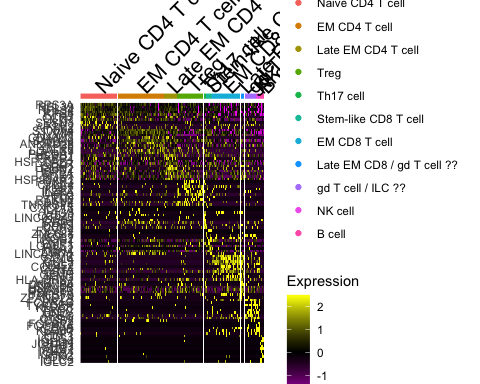
## Calculating cluster Late EM CD8 / gd T cell ??

## Calculating cluster gd T cell / ILC ??

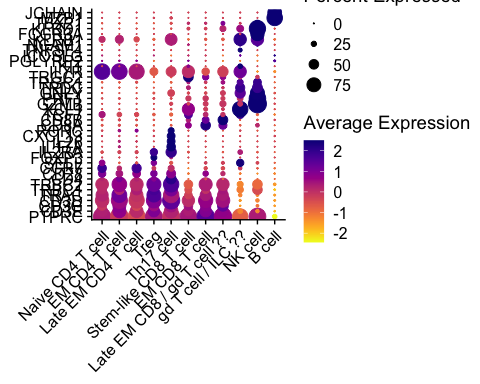
## Calculating cluster NK cell

## Calculating cluster B cell

top10markers <- cluster\_markers %>%  
 group\_by(cluster) %>%  
 top\_n(n = 10, wt = avg\_log2FC)  
  
DoHeatmap(NewLymphocyteClusters, features = top10markers$gene)



DotPlot\_scCustom(seurat\_object = NewLymphocyteClusters, features = genes, flip\_axes = T, x\_lab\_rotate = T)



save(NewLymphocyteClusters, file = "NewLymphocyteClusters.Rdata")

CONCLUSION

Upon expanding this dataset, we were able to go from 5 clusters to 11 clusters (“Naive CD4 T cell”, “EM CD4 T cell”, “Late EM CD4 T cell”, “Treg”, “Th17 cell”, “Stem-like CD8 T cell”, “EM CD8 T cell”, “Late EM CD8 / gd T cell ??”, “gd T cell / ILC ??”, “NK cell”, “B cell”). There may be gd T cells and ILCs mixed within the same cluster because the ILC cluster seems to have cells expressing genes that represent gd T cells (TRDC, TRGC2, TRGC1) when they should not be positive for these genes. Moreover, gd T cells may be present in the Late EM CD8 T cell cluster as well due to similar functional gene signatures. In literature, these cell types have been shown to be quite similar, so we would need a dataset with lots more cells to make a more concrete verification that those are the correctly named clusters.

Ultimately, there are now 11 cell types distinguished in the lymphocyte dataset. Researchers can use this new information to generate hypotheses about genes of interest in more specific cell types.

Another method for annotating cell types may be to use the online source, Azimuth, at <https://azimuth.hubmapconsortium.org/>. It’s super quick and easy to use because you just upload your data to their reference datasets that can then name the clusters, however, there are very limited reference datasets available. In this case, there is no human skin dataset on the site that we can refer to.

CELLxGENE (<https://cellxgene.cziscience.com/>) is another useful source where you can find single-cell data and explore gene expression across many tissues/cell types. It allows you to download and integrate data as well.

```