Tabula Muris Senis Myeloid

Domien Vanneste

2024-12-06 12:01:11 +0100

Contents

Load packages

```
suppressMessages({
    library(Seurat)
    library(ggplot2)
    library(reticulate)
    library(sceasy)
    library(ggsci)
    library("scales")
    library(ggrastr)
    library(OmnipathR)
    library(decoupleR)
    library(dplyr)
    library(tibble)
    library(tidyr)
    library(patchwork)
})
```

Download Tabula Muris Senis scanpy object from AWS server

```
~$ aws s3 cp --no-sign-request s3://czb-tabula-muris-senis/Data-objects/tabula-muris-senis-bbknn-proces
download: s3://czb-tabula-muris-senis/Data-objects/tabula-muris-senis-bbknn-processed-official-annotati
```

Convert Scanpy object to Seurat object

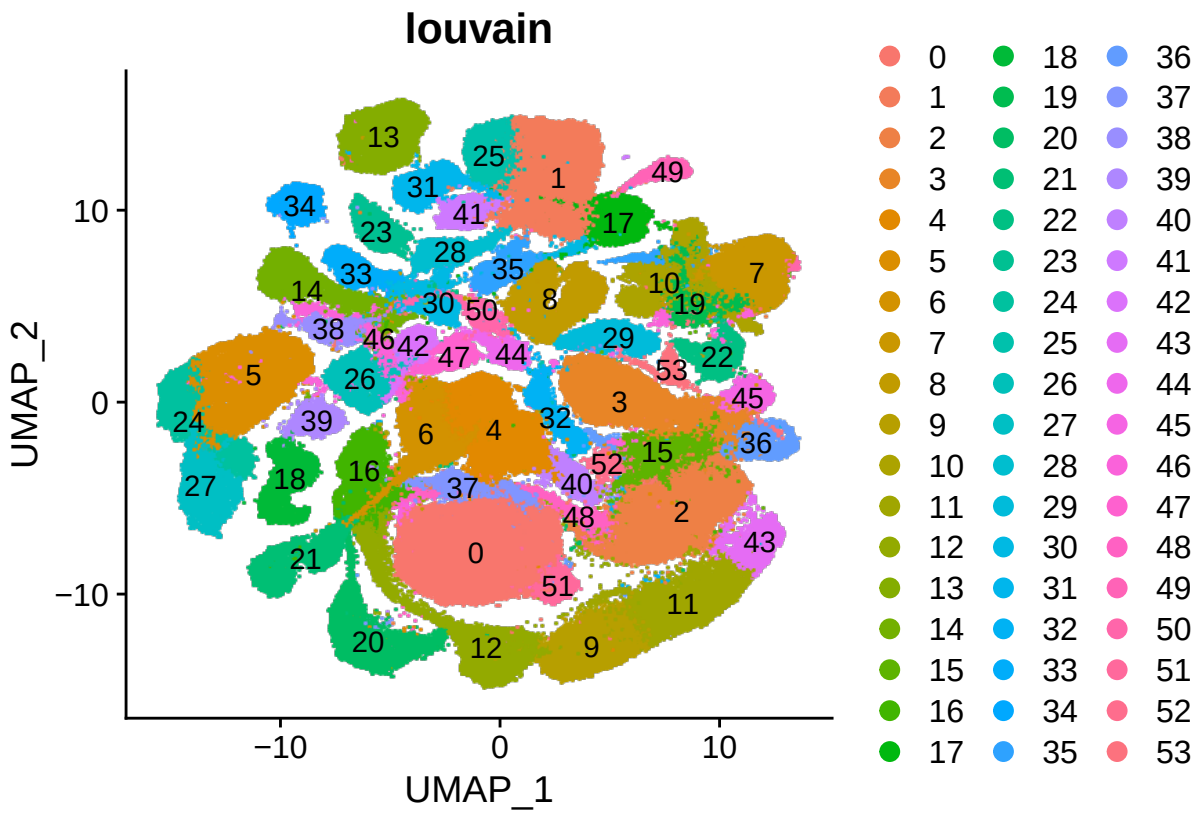
```
sceasy::convertFormat("tabula-muris-senis-bbknn-processed-official-annotations.h5ad",
    from = "anndata", to = "seurat", outFile = "tbms.rds")
```

Load Seurat object

```
tbms <- readRDS(file = "tbms.rds")
```

```
DimPlot(tbms, group.by = "louvain", label = T, raster = T)
```

```
## Rasterizing points since number of points exceeds 100,000.
## To disable this behavior set ‘raster=FALSE‘
```

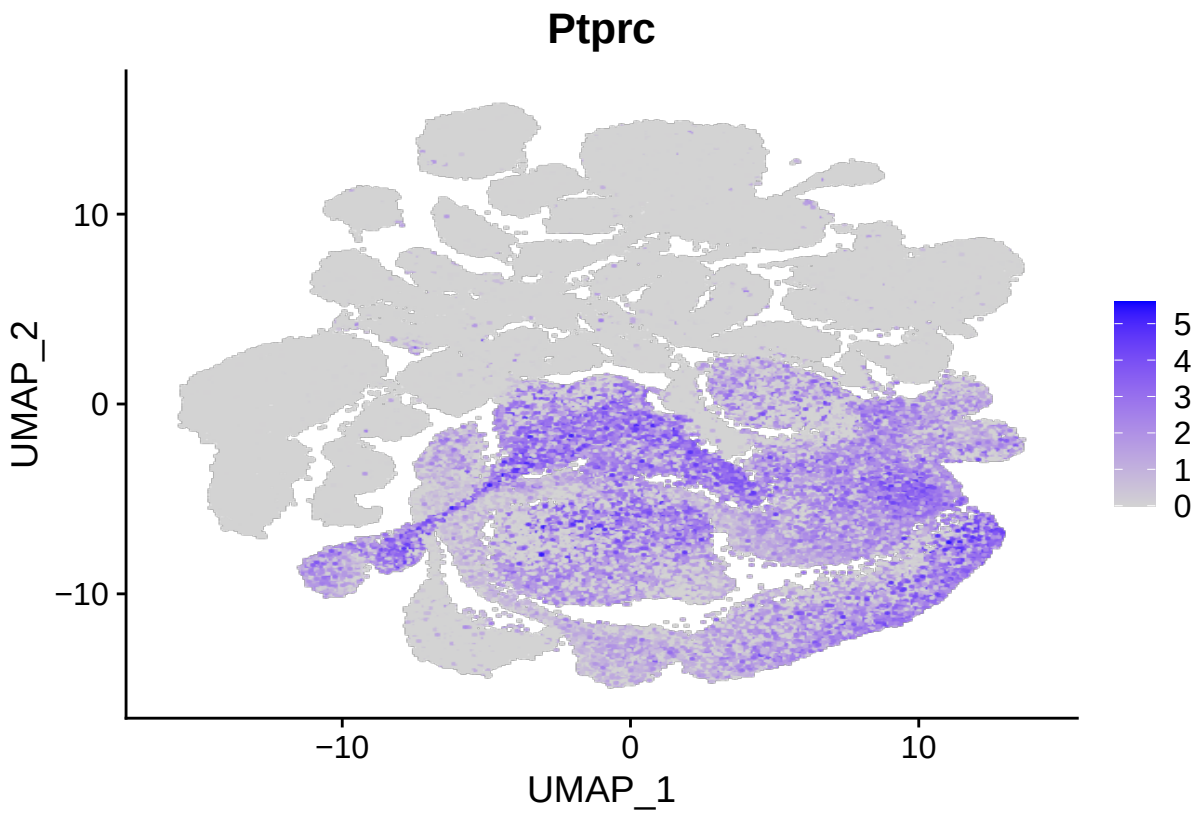


Subset myeloid cells

Myeloid cell clusters were identified based on the combined expression of Ptprc (CD45) anf Lyz2 (LysM).

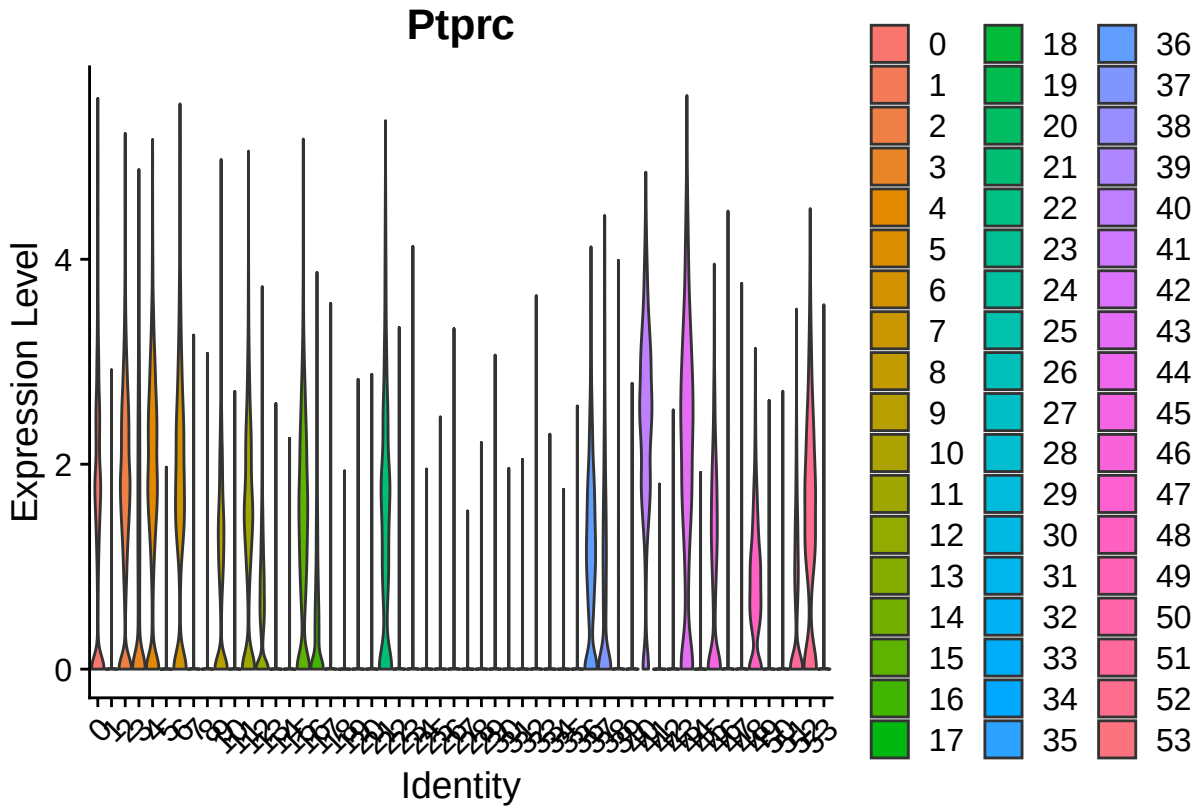
```
FeaturePlot(tbms, features = "Ptprc", raster = T)
```

```
## Rasterizing points since number of points exceeds 100,000.
## To disable this behavior set ‘raster=FALSE‘
```



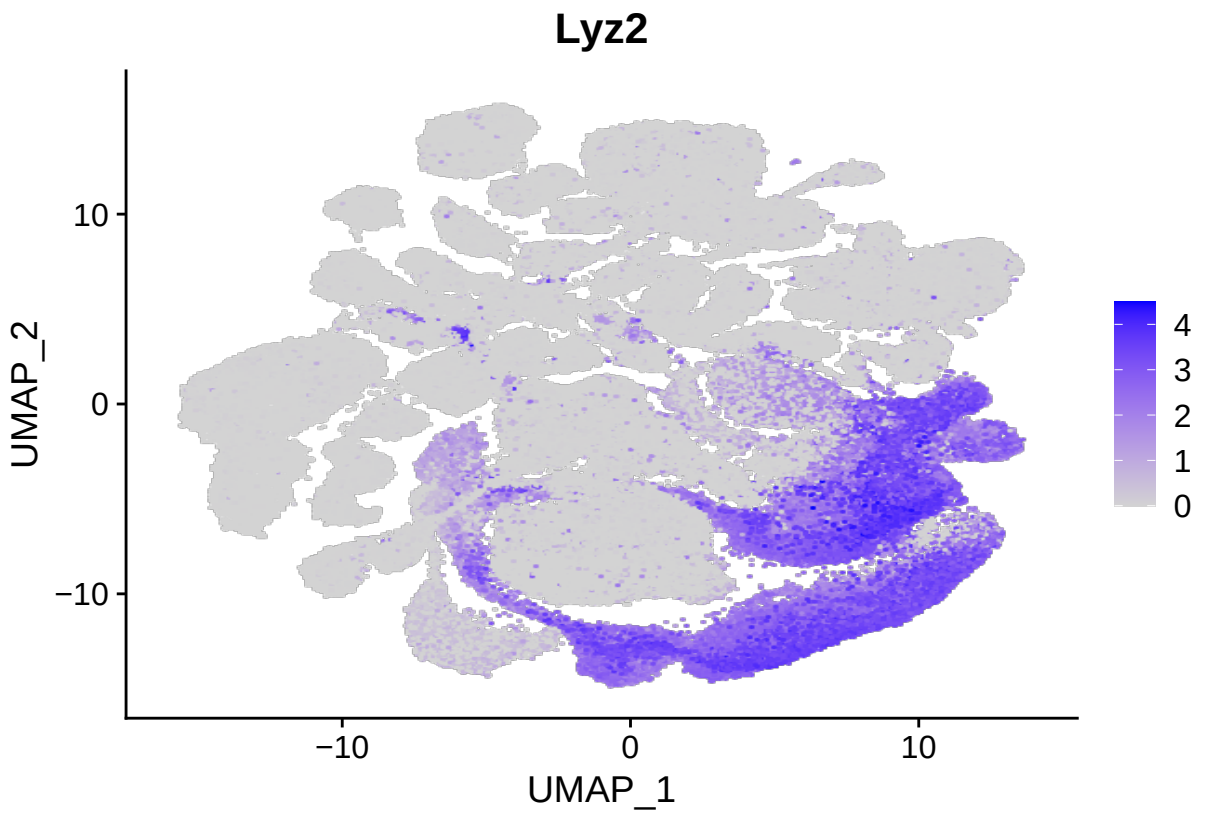

```
VlnPlot(tbms, features = "Ptprc", group.by = "louvain", pt.size = 0)
```

```
## Rasterizing points since number of points exceeds 100,000.
## To disable this behavior set ‘raster=FALSE‘
```



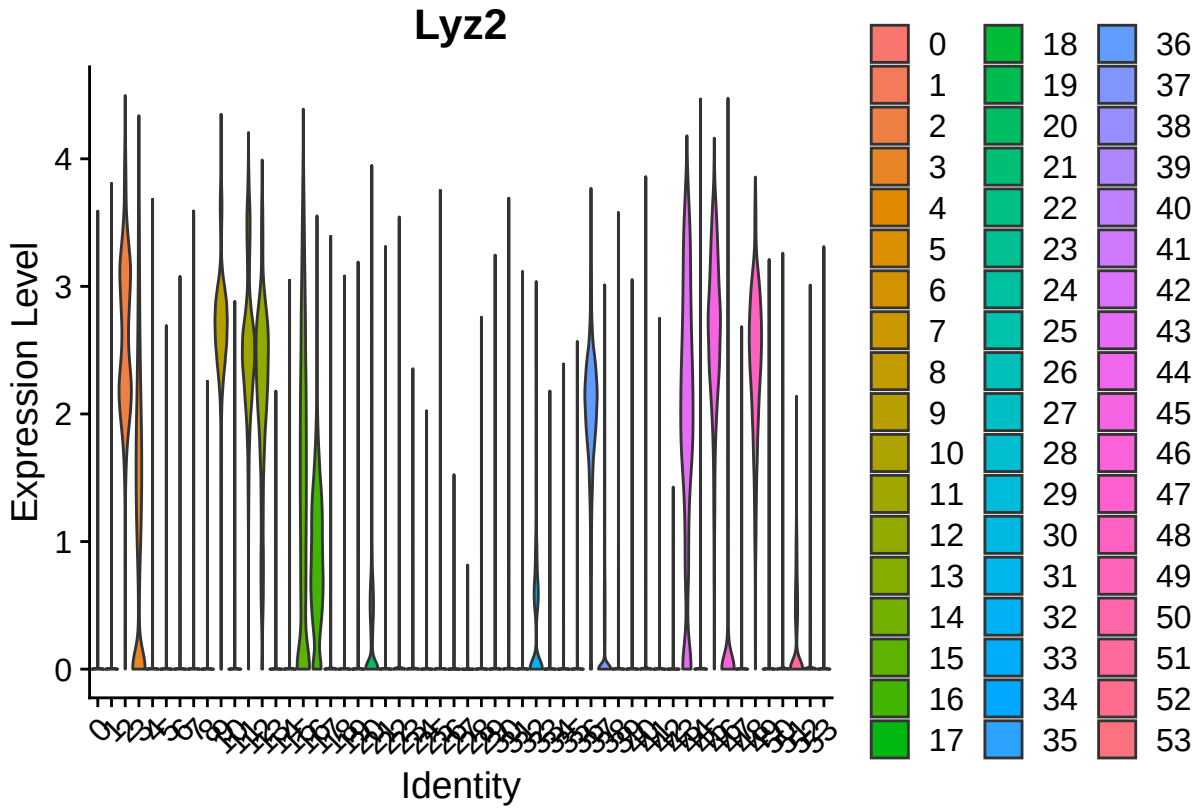

```
FeaturePlot(tbms, features = "Lyz2", raster = T)
```

```
## Rasterizing points since number of points exceeds 100,000.
## To disable this behavior set 'raster=FALSE'
```



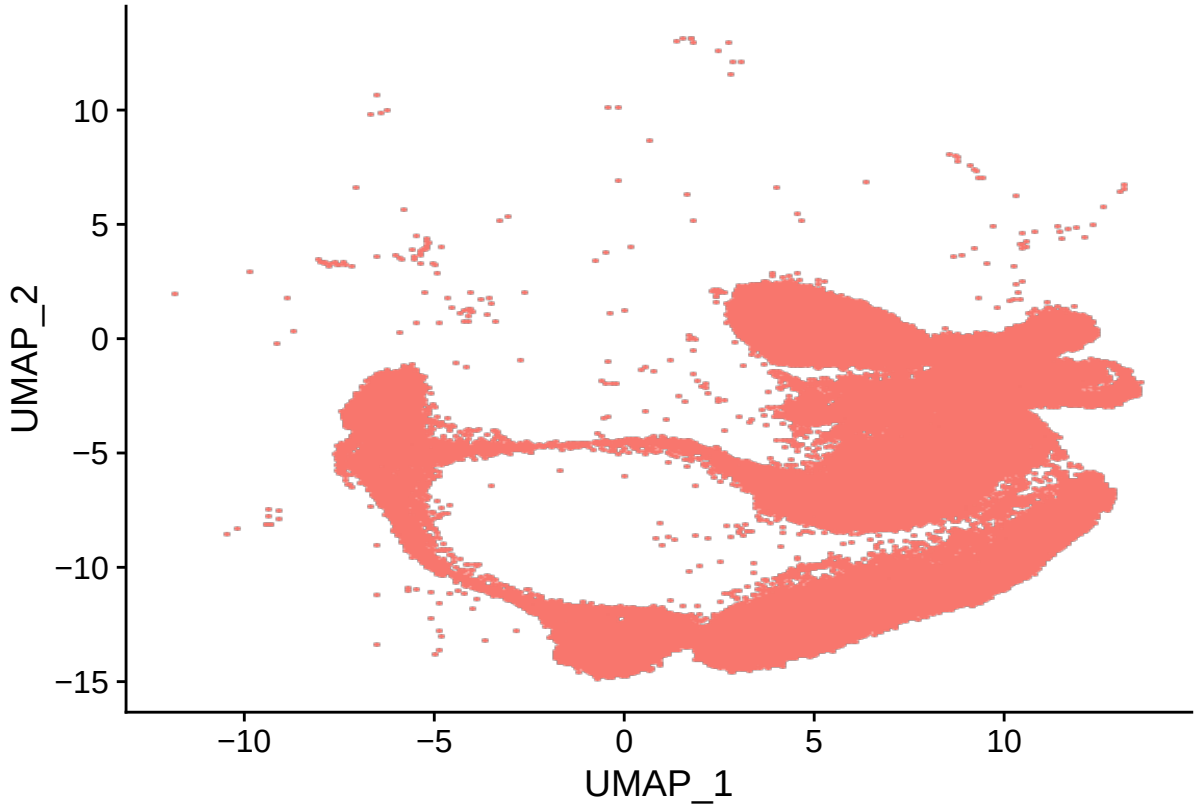

```
VlnPlot(tbms, features = "Lyz2", group.by = "louvain", pt.size = 0)
```

```
## Rasterizing points since number of points exceeds 100,000.
## To disable this behavior set ‘raster=FALSE‘
```




```
Myeloid_Cells <- c(2, 3, 9, 11, 12, 15, 16, 36, 43, 45, 48, 52)

tbms_myeloid <- subset(tbms, subset = louvain %in% Myeloid_Cells)
```

```
DimPlot(tbms_myeloid, raster = T) + theme(legend.position = "none")
```



Normalization and scaling of myeloid cells

```
tbms_myeloid <- FindVariableFeatures(tbms_myeloid, selection.method = "vst",
    nfeatures = 2000)
all.genes <- rownames(tbms_myeloid)
tbms_myeloid <- ScaleData(tbms_myeloid, features = all.genes)
```

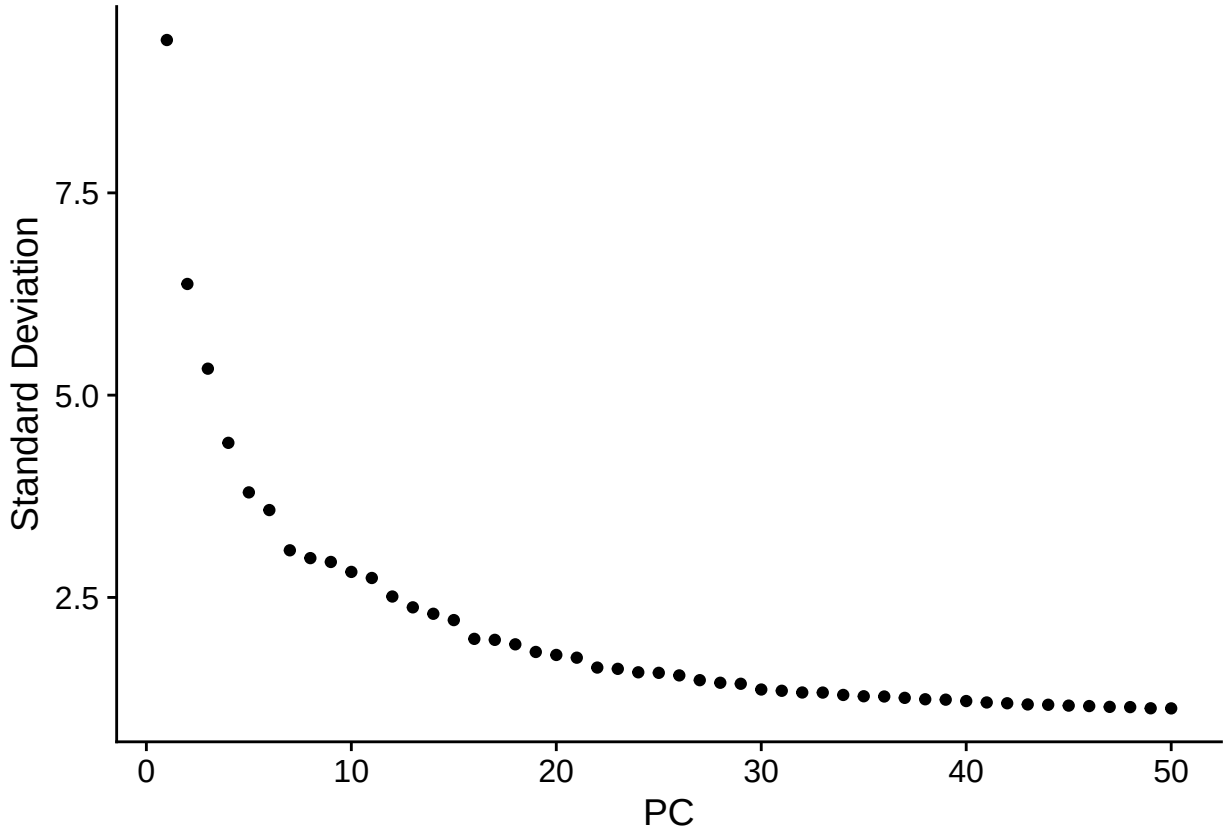
```
## Centering and scaling data matrix
```

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

```
## PC_ 1
## Positive:  S100a8, S100a9, Hp, Lcn2, Pglyrp1, 1100001G20Rik, Gsr, Ifitm6, Ngp, Ly6c2
##     Sepx1, 1810033B17Rik, Camp, Slpi, Trem3, Lrg1, Cd177, Ltf, Chi3l3, Retnlg
##     Glrx, Pygl, Ltb4r1, Adpgk, Cebpe, AA467197, Mmp9, Itgb2l, Mmp8, Chi3l1
## Negative:  Lgmn, Csf1r, C1qb, P2ry12, C1qc, Ctss, C1qa, Tmem119, Olfml3, Hexb
##     Gpr34, Cx3cr1, Abhd12, Itgb5, Fcrls, Trem2, Siglech, P2ry13, Cd68, Ctsd
##     Pld4, Slc2a5, Adap2, Selplg, Hpgds, Fam105a, Slco2b1, Ptgs1, Lpcat2, Golm1
## PC_ 2
## Positive:  Ngp, Camp, Ltf, Lcn2, S100a9, Pglyrp1, Cd177, S100a8, Adpgk, Cebpe
##     Tmem119, 1100001G20Rik, Olfml3, Itgb2l, Chi3l1, P2ry12, Slc2a5, Gpr34, Siglech, Ncf1
##     Lrg1, 4632428N05Rik, P2ry13, AA467197, Syngr1, Hexb, Trem3, Golm1, Lpcat2, Gal3st4
```

```
## Negative:  Gm11428, Clec4a1, Ms4a6c, Clec4n, Ccl6, S100a4, Ccl9, Msr1, Pf4, Sdc3
##     Cfp, Clec7a, Eef1a1, Folr2, Vsig4, Mrc1, Cd5l, Ms4a7, Il1b, Clec4a3
##     Fcna, Clec4b1, Bcl2a1d, Ccdc109b, LOC100038947, H2-DMb1, Nr1h3, Bcl2a1b, Mpeg1, Clec4f
## PC_ 3
## Positive:  Mmp9, C5ar1, Mmp8, Mxd1, Cxcr2, Retnlg, Pilra, Fpr2, Fpr1, Il1f9
##     Hdc, Gp49a, Lrg1, Ccr1, Slfn1, Clec4d, Arg2, Lilrb4, Slfn4, Ifitm6
##     Cd300lf, Chi3l1, Gm10872, Ceacam10, G0s2, Itgam, Fcgr3, Ccl6, Mrgpra2b, 1810033B17Rik
## Negative:  Eef1a1, Prtn3, Mpo, Ctsg, Rab38, BC005764, Syce2, Fam69b, Angpt1, Nme4
##     Ms4a3, Cdca5, Elane, Prss57, Cdc45, Cdca2, Bex6, 2610002D18Rik, Sdsl, Spns3
##     Dctd, Slc22a3, Mpl, Nuf2, Sh2d5, Trip13, Flt3, Gins1, F2rl3, Shcbp1
## PC_ 4
## Positive:  Clec4f, Vsig4, Cd5l, Clec1b, Folr2, Fcna, Fabp7, Cd207, Cxcl13, Timd4
##     Ccl24, Pf4, Il18bp, Gdf15, AI606473, Slc40a1, Ccr3, Nr1h3, Gbp1, Clec4n
##     Smc1b, C6, Ms4a7, Sdc3, Cebpe, Fcnb, St3gal5, Clec12a, Ccl8, Mrc1
## Negative:  S100a4, Ccr2, Apoc2, LOC100038947, Emilin2, Gm9733, F10, Ccdc109b, Fgr, Ms4a6c
##     Sirpb1b, Atp1a3, Selplg, Ccl6, Gda, Clec4e, Amica1, Cd244, Slc2a6, Rnf149
##     Naaa, F13a1, Tnfrsf1b, Lilra6, Galnt9, Gpr141, Ccl9, Pld4, Slfn1, Slc16a3
## PC_ 5
## Positive:  Angpt1, Rab38, Mpl, Ifitm1, Sdsl, Fam69b, Slc22a3, Nme4, Cxcr2, F2rl3
##     Flt3, Mxd1, Il1f9, Gm5111, P2ry14, Arg2, Dctd, Myl10, Gbx2, Hdc
##     Adam8, Clec4d, Amica1, Dapp1, Itga2b, Gm10872, Sh2d5, Otos, Mmp9, Il1r2
## Negative:  Chi3l3, Lyz1, Ly6c2, Ms4a6c, F10, S100a4, F13a1, Ccr2, Sepx1, Clec4a2
##     Fcnb, Apoc2, Lst1, Hp, 1700020L24Rik, Cdkn3, Igsf6, Ccdc109b, Cebpe, Clec4a3
##     Gsr, Clec4a1, Clec12a, Gpr141, Elane, Orm1, Mogat2, Tifab, Camp, Ifitm6
```

```
ElbowPlot(tbms_myeloid, ndims = 50)
```



Clustering of myeloid cells

```
tbms_myeloid <- FindNeighbors(tbms_myeloid, dims = 1:7)
```

```
## Computing nearest neighbor graph
```

```
## Computing SNN
```

```
tbms_myeloid <- FindClusters(tbms_myeloid, resolution = 0.35)
```

```
## Modularity Optimizer version 1.3.0 by Ludo Waltman and Nees Jan van Eck
##
## Number of nodes: 87136
## Number of edges: 2642739
##
## Running Louvain algorithm...
## Maximum modularity in 10 random starts: 0.9492
## Number of communities: 18
## Elapsed time: 36 seconds
```

```
tbms_myeloid <- RunUMAP(tbms_myeloid, dims = 1:7)
```

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

```
## 12:05:39 UMAP embedding parameters a = 0.9922 b = 1.112
```

```
## 12:05:39 Read 87136 rows and found 7 numeric columns
```

```
## 12:05:39 Using Annoy for neighbor search, n_neighbors = 30
```

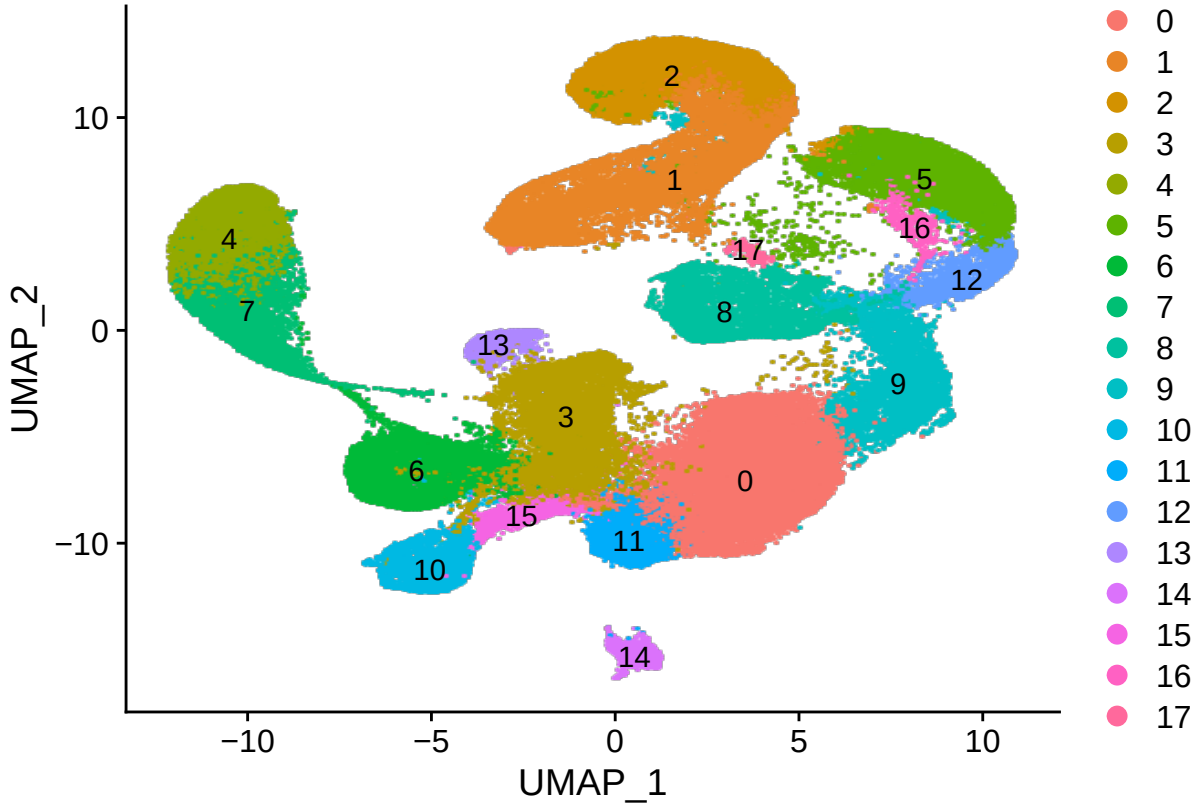
```
## 12:05:39 Building Annoy index with metric = cosine, n_trees = 50
```

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

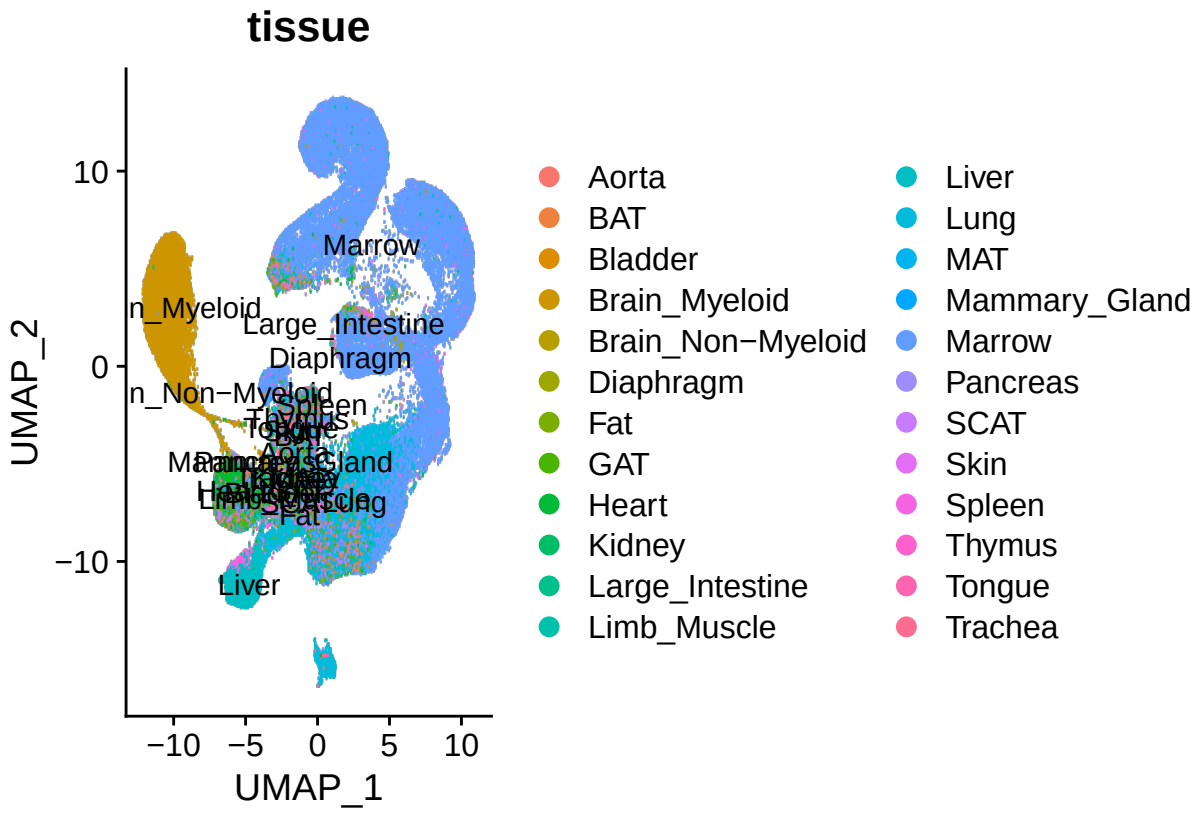
```
## [----|----|----|----|----|----|----|----|----|----|
```

```
## **************************************************|
## 12:05:45 Writing NN index file to temp file /tmp/Rtmp7dH98F/file13f5274c195fca
## 12:05:45 Searching Annoy index using 1 thread, search_k = 3000
## 12:06:12 Annoy recall = 100%
## 12:06:12 Commencing smooth kNN distance calibration using 1 thread with target n_neighbors = 30
## 12:06:14 Initializing from normalized Laplacian + noise (using irlba)
## 12:06:18 Commencing optimization for 200 epochs, with 3423834 positive edges
## 12:06:40 Optimization finished
```

```
DimPlot(tbms_myeloid, label = T, raster = T)
```




```
DimPlot(tbms_myeloid, group.by = "tissue", label = T, raster = T)
```



Removal of contaminating cells

```
cluster16.markers <- FindMarkers(tbms_myeloid, ident.1 = 16,
    only.pos = TRUE)
```

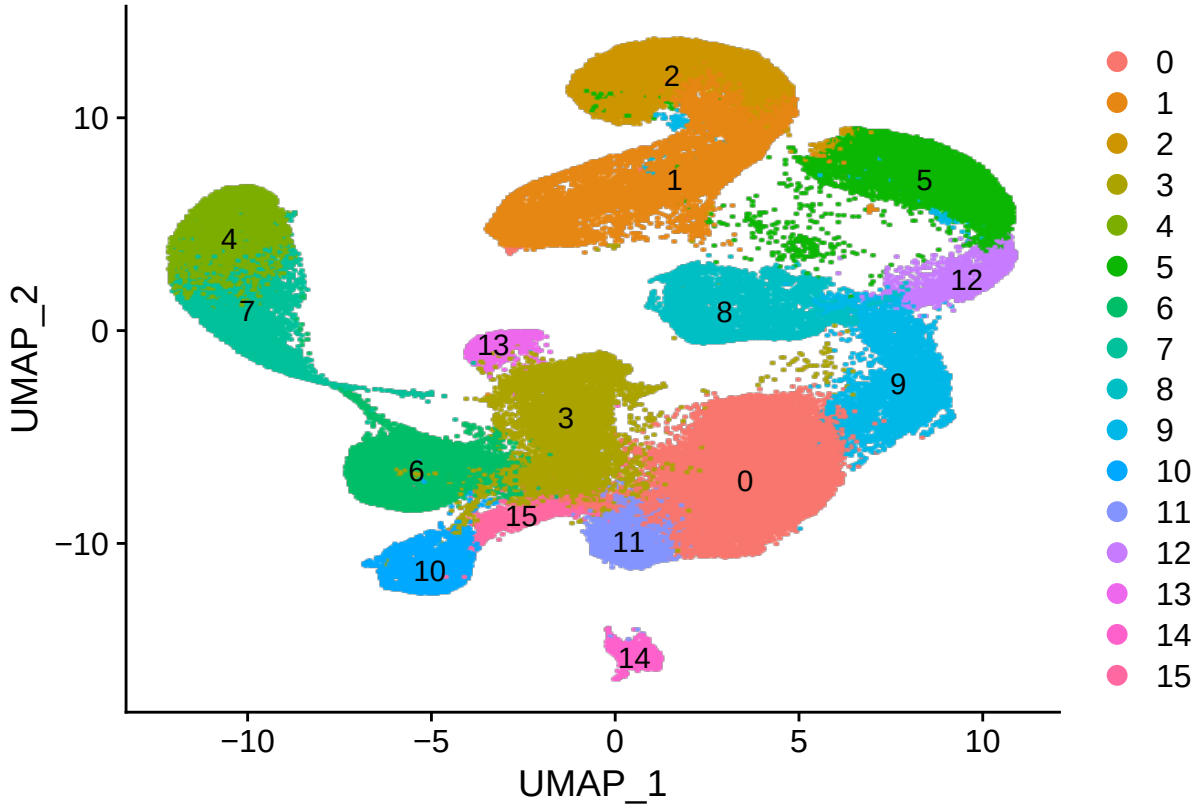
Cluster 16 are proliferating cells

```
cluster17.markers <- FindMarkers(tbms_myeloid, ident.1 = 17,
    only.pos = TRUE)
```

Cluster 17 are mast cells

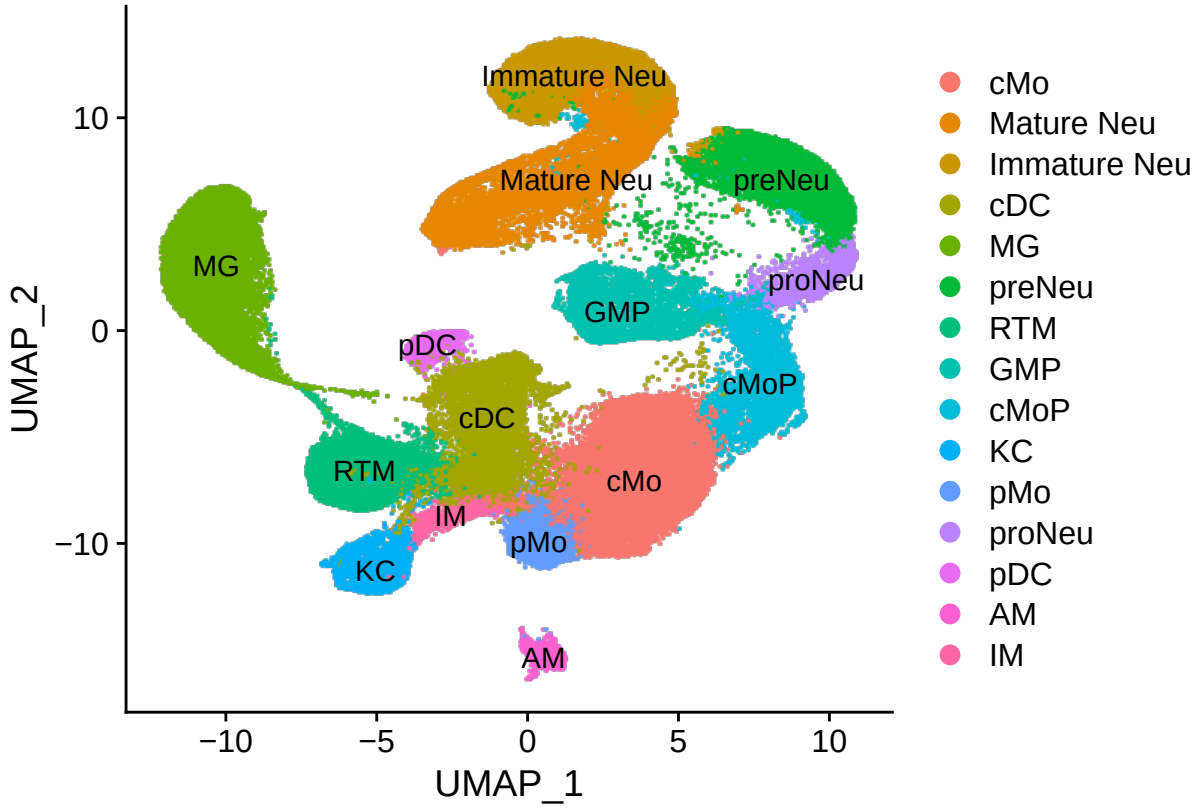
```
tbms_myeloid <- subset(tbms_myeloid, idents = c(16, 17), invert = T)
```

```
DimPlot(tbms_myeloid, label = T, raster = T)
```

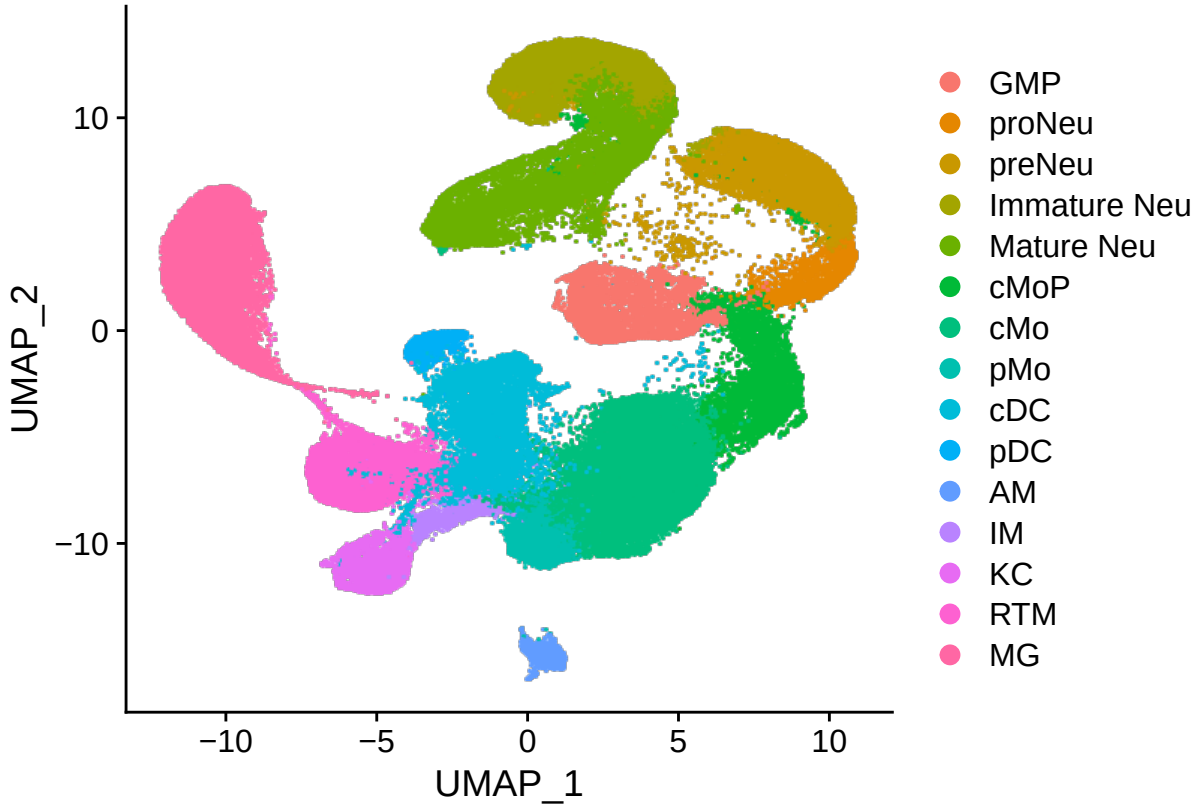


Assigning celltypes to clusters

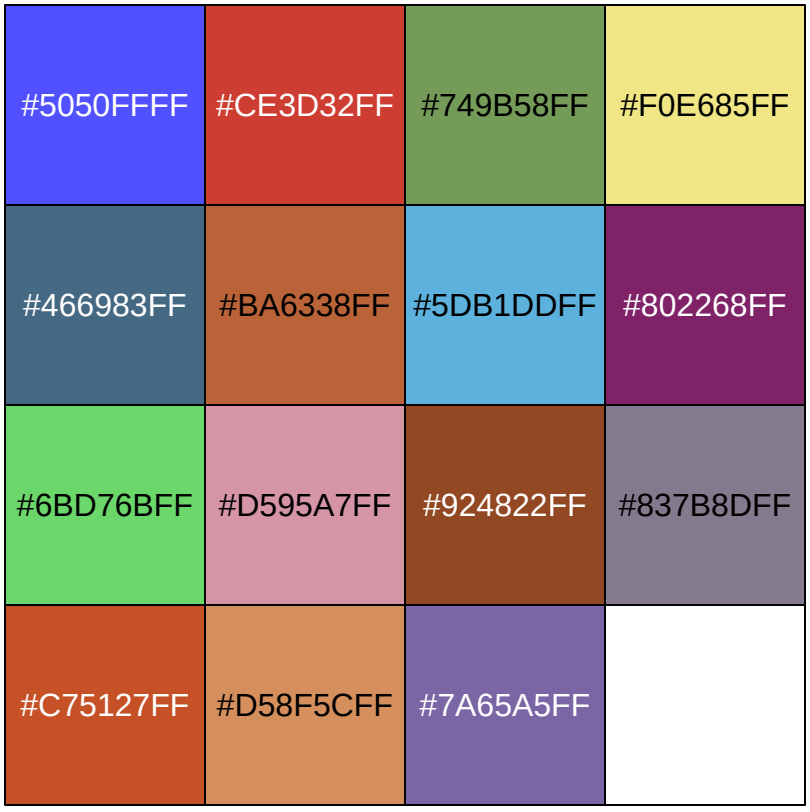
```
tbms_myeloid$cell_type <- factor(Idents(tbms_myeloid), labels = c("cMo",
    "Mature Neu", "Immature Neu", "cDC", "MG", "preNeu", "RTM",
    "MG", "GMP", "cMoP", "KC", "pMo", "proNeu", "pDC", "AM",
    "IM"))

tbms_myeloid$cell_type <- factor(tbms_myeloid$cell_type, levels = c("cMo",
    "Mature Neu", "Immature Neu", "cDC", "MG", "preNeu", "RTM",
    "GMP", "cMoP", "KC", "pMo", "proNeu", "pDC", "AM", "IM"))

Idents(tbms_myeloid) <- "cell_type"

DimPlot(tbms_myeloid, label = T, raster = T)
```



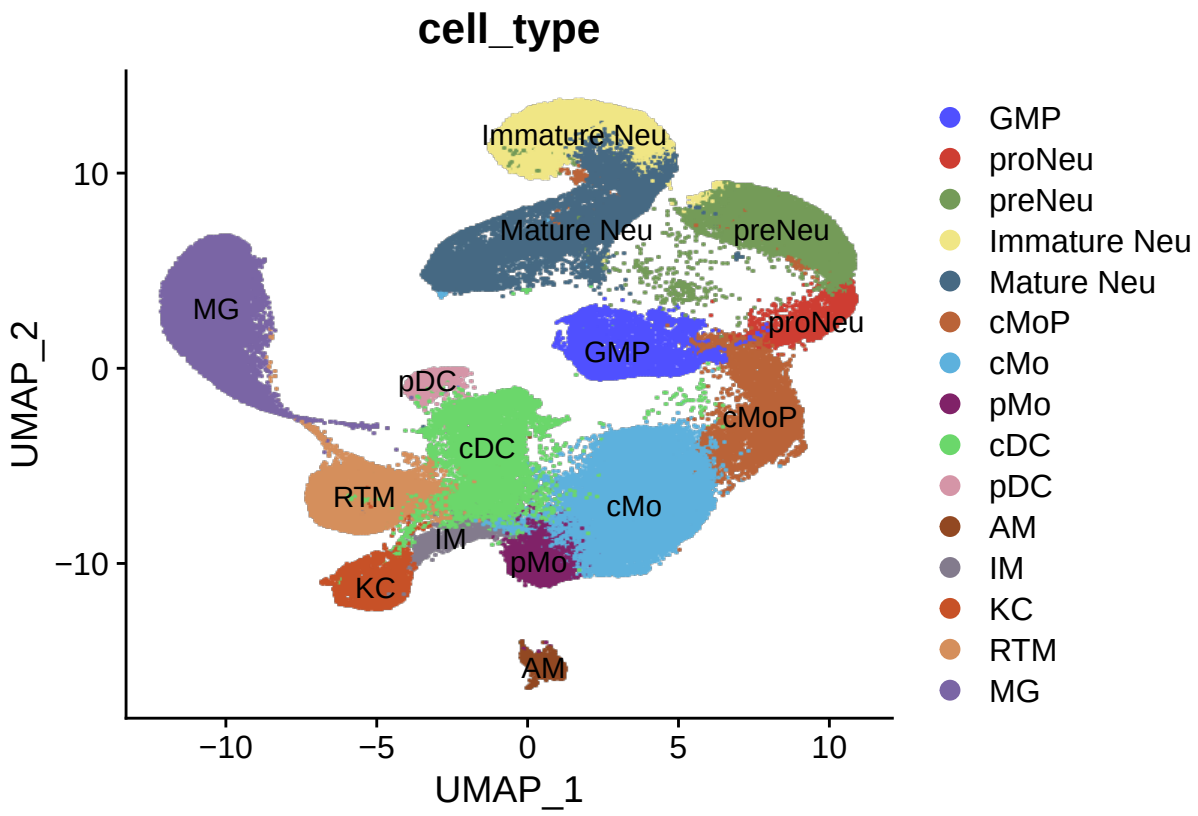

```
tbms_myeloid$cell_type <- factor(tbms_myeloid$cell_type, levels = c("GMP",
    "proNeu", "preNeu", "Immature Neu", "Mature Neu", "cMoP",
    "cMo", "pMo", "cDC", "pDC", "AM", "IM", "KC", "RTM", "MG"))

Idents(tbms_myeloid) <- "cell_type"

DimPlot(tbms_myeloid, raster = T)
```



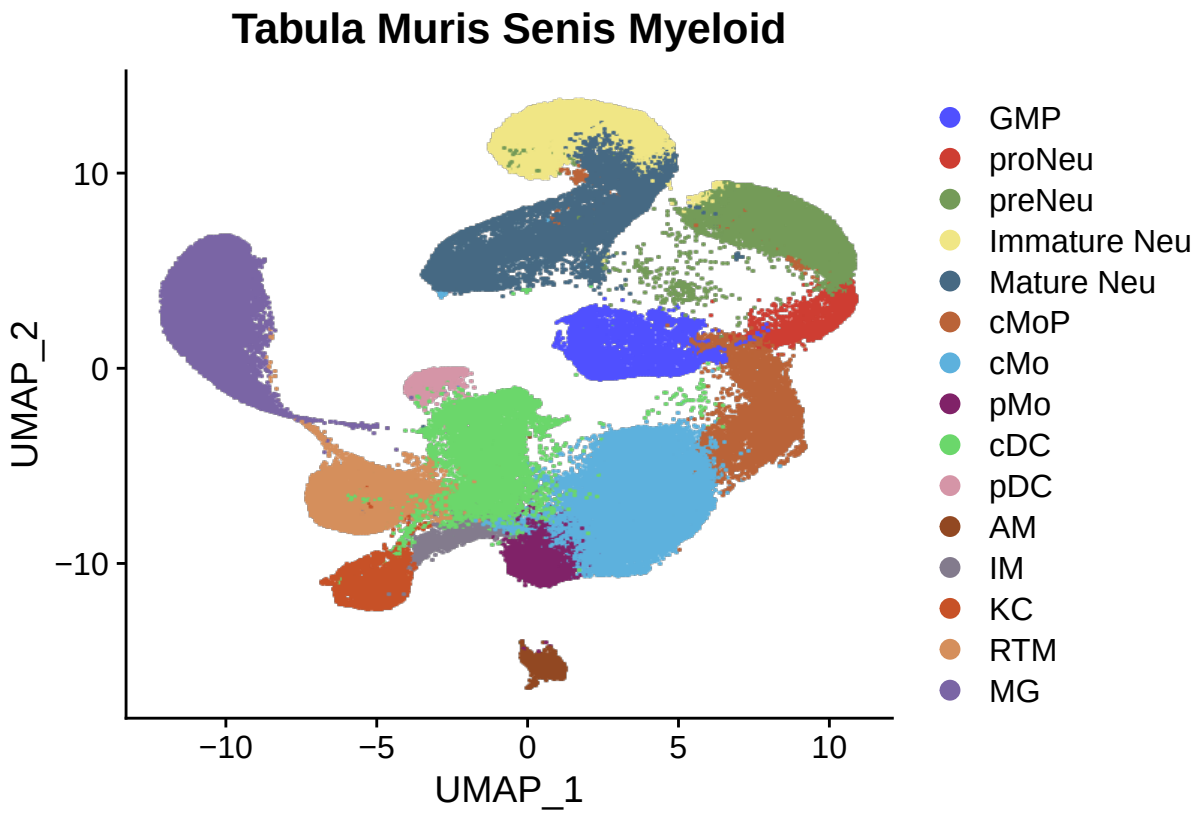

```
show_col(pal_igv("default")(15))
```




```
cols <- c(pal_igv("default")(15))

DimPlot(tbms_myeloid, label = T, cols = cols, group.by = "cell_type",
    raster = T)
```

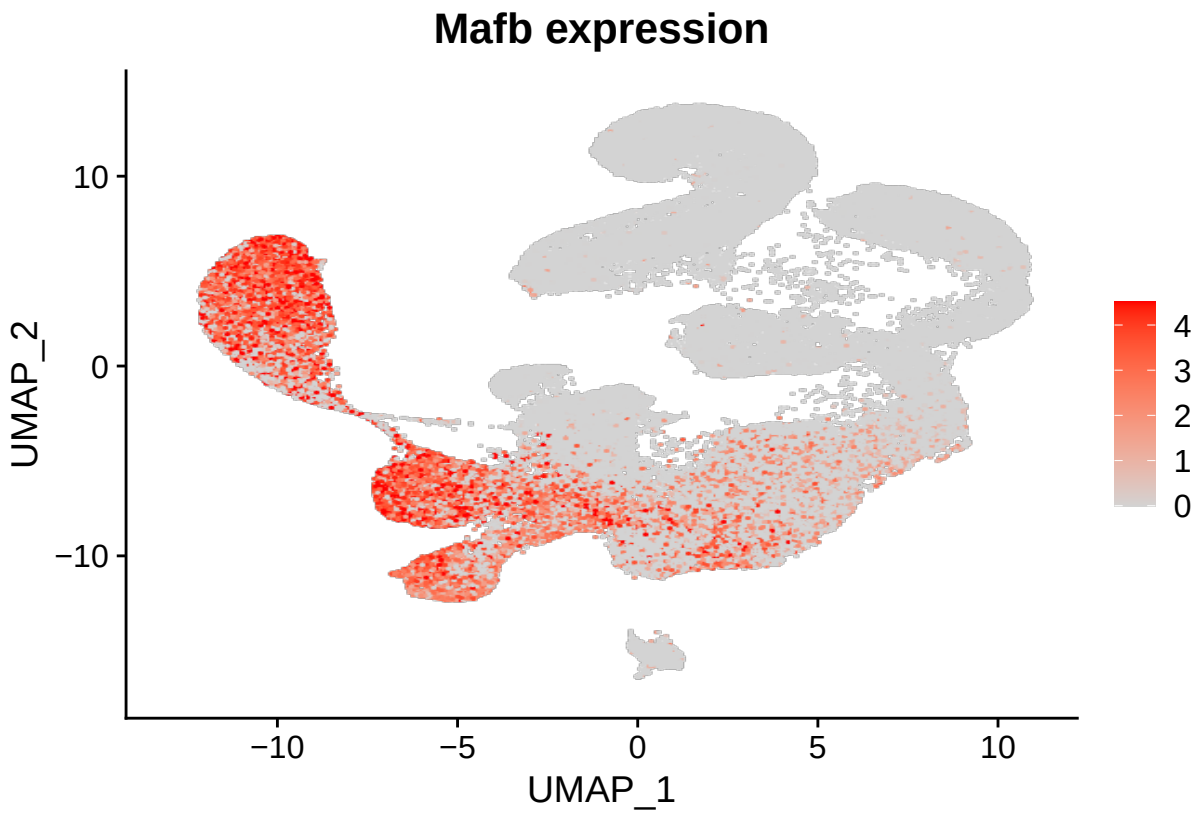



```
DimPlot(tbms_myeloid, cols = cols, group.by = "cell_type", raster = T) +
    ggtitle("Tabula Muris Senis Myeloid")
```

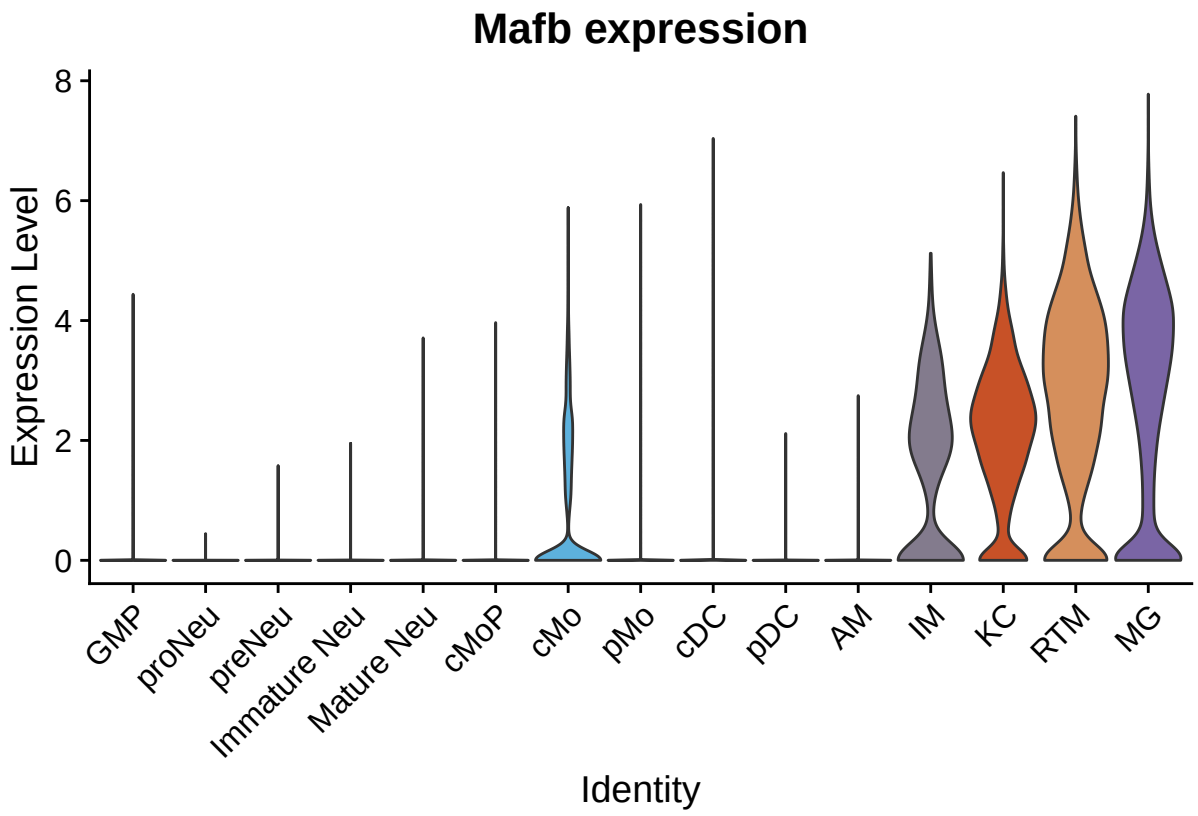


Viualize Mafb expression

```
FeaturePlot(tbms_myeloid, features = "Mafb", max.cutoff = "q90",
    cols = c("lightgrey", "red"), raster = T) + ggtitle("Mafb expression")
```




```
VlnPlot(tbms_myeloid, features = "Mafb", pt.size = 0, cols = cols) +
    NoLegend() + ggtitle("Mafb expression")
```



Calculate and viualize MafB activity with decoupleR

Load CollecTRI network

CollecTRI is a comprehensive resource containing a curated collection of TFs and their transcriptional targets.

```
net <- get_collectri(organism = "mouse", split_complexes = FALSE)
net
```

```
## # A tibble: 38,665 x 3
##    source target   mor
##    <chr>  <chr>  <dbl>
##  1 Myc    Tert       1
##  2 Spi1   Bglap      1
##  3 Spi1   Bglap3     1
##  4 Spi1   Bglap2     1
##  5 Smad3  Jun        1
##  6 Smad4  Jun        1
##  7 Stat5a Il2        1
##  8 Stat5b Il2        1
##  9 Rela   Fas        1
## 10 Wt1    Nr0b1      1
## # i 38,655 more rows
```

Activity inference with Univariate Linear Model (ULM)

To infer TF enrichment scores we will run the Univariate Linear Model (ulm) method. For each sample in our dataset (mat) and each TF in our network (net), it fits a linear model that predicts the observed gene expression based solely on the TF's TF-Gene interaction weights. Once fitted, the obtained t-value of the slope is the score. If it is positive, we interpret that the TF is active and if it is negative we interpret that it is inactive.

```
# Extract the normalized log-transformed counts
mat <- as.matrix(tbms_myeloid@assays$RNA@data)
```

```
## Warning in asMethod(object): sparse->dense coercion: allocating vector of size
## 13.0 GiB
```

```
# Run ulm
acts <- run_ulm(mat = mat, net = net, .source = "source", .target = "target",
    .mor = "mor", minsize = 5)

acts
```

```
## # A tibble: 60,767,320 x 5
##    statistic source condition                    score  p_value
##    <chr>     <chr>  <chr>                        <dbl>    <dbl>
##  1 ulm       Abl1   10X_P4_0_AAGGCAGGTGGTTTCA-1-0  2.65 0.00797
##  2 ulm       Abl1   10X_P4_0_AAGGCAGTCTGTACGA-1-0  2.43 0.0150
##  3 ulm       Abl1   10X_P4_0_ACATACGAGAATCTCC-1-0  2.73 0.00643
##  4 ulm       Abl1   10X_P4_0_ACGATACTCCTTGGTC-1-0  1.20 0.230
##  5 ulm       Abl1   10X_P4_0_AGCATACCAGATGAGC-1-0  1.42 0.154
##  6 ulm       Abl1   10X_P4_0_AGGCCGTCATTACGAC-1-0  2.88 0.00393
##  7 ulm       Abl1   10X_P4_0_CATTATCGTCGGATCC-1-0  2.78 0.00549
##  8 ulm       Abl1   10X_P4_0_CGATGGCTCACCACCT-1-0  1.51 0.130
##  9 ulm       Abl1   10X_P4_0_CGGACGTGTGTCGCTG-1-0  3.36 0.000768
## 10 ulm       Abl1   10X_P4_0_CTTACCGTCTCGGACG-1-0  2.56 0.0104
## # i 60,767,310 more rows
```

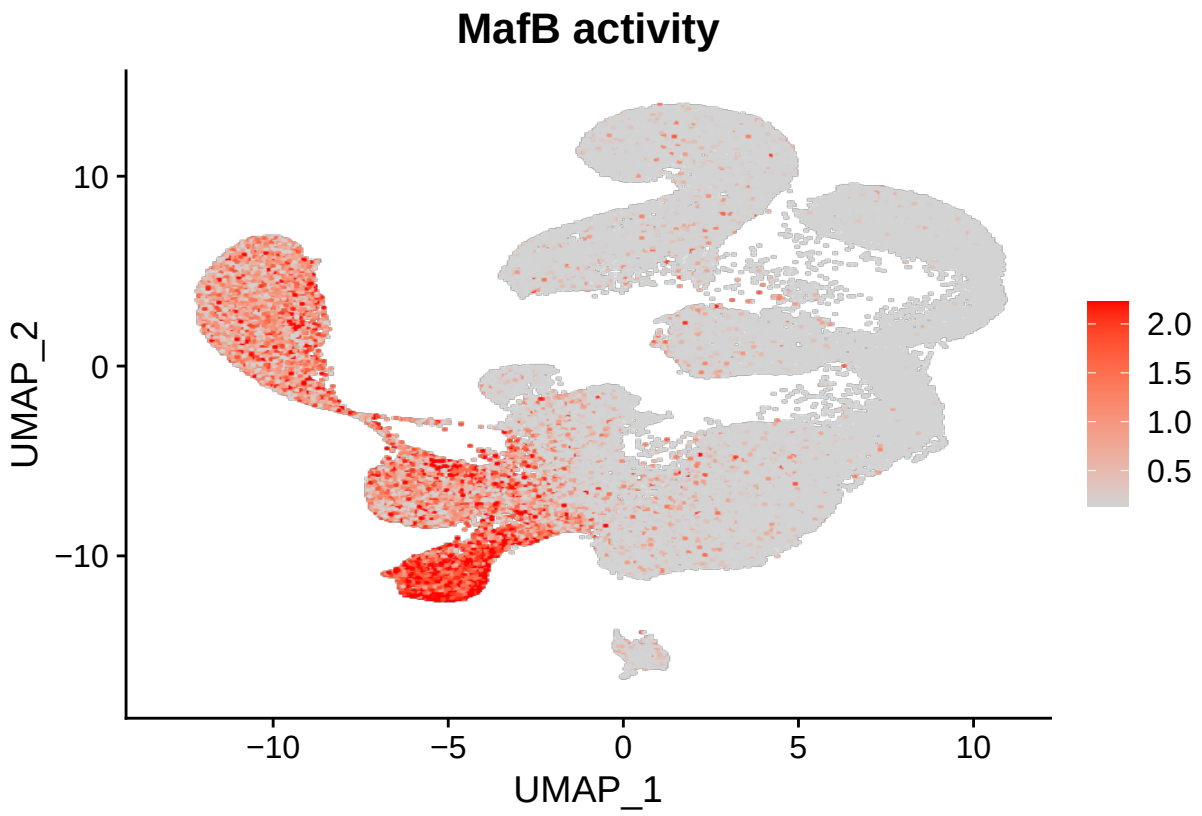
Visualize

```
# Extract ulm and store it in tfsulm in seurat object
tbms_myeloid[["tfsulm"]] <- acts %>%
    pivot_wider(id_cols = "source", names_from = "condition",
        values_from = "score") %>%
    column_to_rownames("source") %>%
    Seurat::CreateAssayObject(.)

# Change assay
DefaultAssay(object = tbms_myeloid) <- "tfsulm"

# Scale the data
tbms_myeloid <- ScaleData(tbms_myeloid)
```

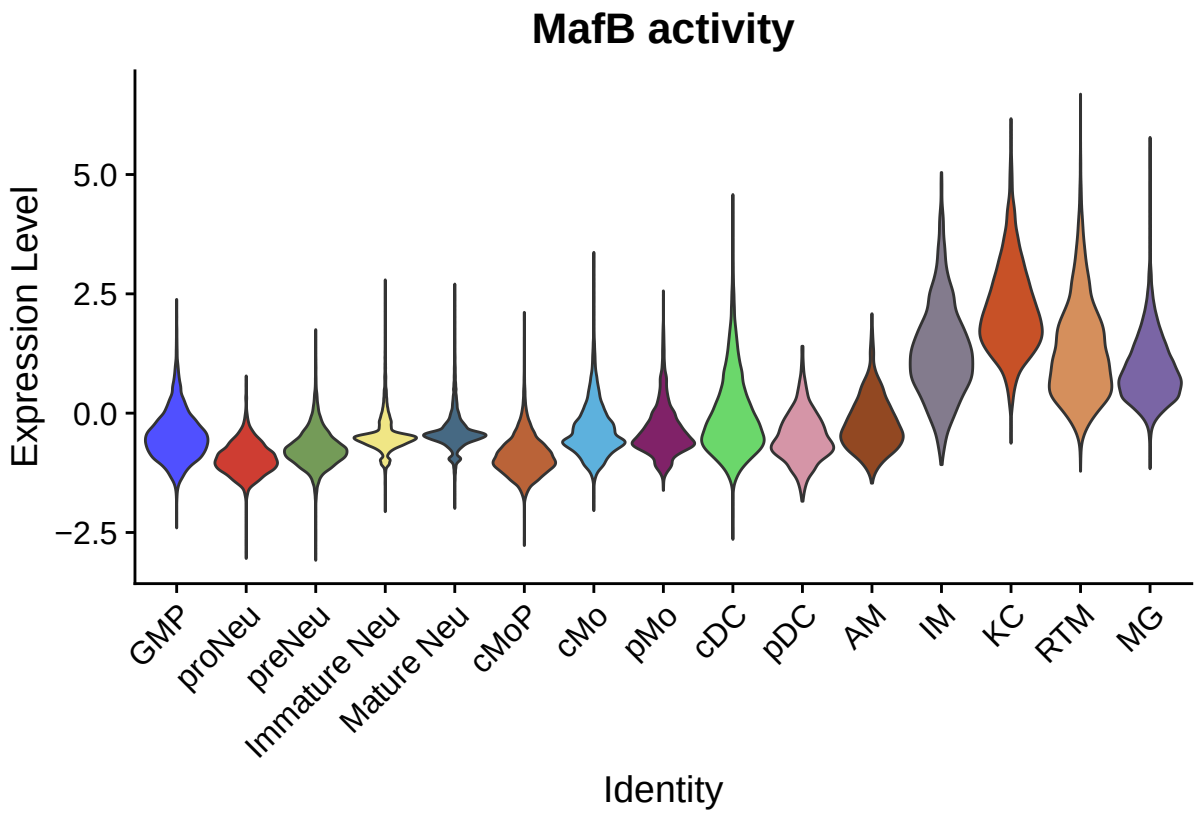
```
## Centering and scaling data matrix
```

```
tbms_myeloid@assays$tfsulm@data <- tbms_myeloid@assays$tfsulm@scale.data
```

```
FeaturePlot(tbms_myeloid, features = "Mafb", min.cutoff = "q10",
    max.cutoff = "q90", cols = c("lightgrey", "red"), raster = T) +
    ggtitle("MafB activity")
```




```
VlnPlot(tbms_myeloid, features = "Mafb", pt.size = 0, cols = cols) +
    NoLegend() + ggtitle("MafB activity")
```



Session information

R sesssion:

```
sessionInfo()
```

```
## R version 4.4.1 (2024-06-14)
## Platform: x86_64-pc-linux-gnu
## Running under: Ubuntu 20.04.6 LTS
##
## Matrix products: default
## BLAS:   /usr/lib/x86_64-linux-gnu/blas/libblas.so.3.9.0
## LAPACK: /usr/lib/x86_64-linux-gnu/lapack/liblapack.so.3.9.0
##
## locale:
##  [1] LC_CTYPE=en_US.UTF-8       LC_NUMERIC=C
##  [3] LC_TIME=fr_BE.UTF-8        LC_COLLATE=en_US.UTF-8
##  [5] LC_MONETARY=fr_BE.UTF-8    LC_MESSAGES=en_US.UTF-8
##  [7] LC_PAPER=fr_BE.UTF-8       LC_NAME=C
##  [9] LC_ADDRESS=C               LC_TELEPHONE=C
## [11] LC_MEASUREMENT=fr_BE.UTF-8 LC_IDENTIFICATION=C
##
## time zone: Europe/Brussels
## tzcode source: system (glibc)
```

```
##
## attached base packages:
## [1] stats     graphics  grDevices utils     datasets  methods   base
##
## other attached packages:
##  [1] patchwork_1.1.1    tidyr_1.3.1        tibble_3.2.1       dplyr_1.1.4
##  [5] decoupleR_2.10.0   OmnipathR_3.15.1   ggrastr_1.0.2      scales_1.3.0
##  [9] ggsci_3.2.0        sceasy_0.0.7       reticulate_1.39.0  ggplot2_3.4.4
## [13] SeuratObject_4.1.3 Seurat_4.3.0
##
## loaded via a namespace (and not attached):
##   [1] RcppAnnoy_0.0.18       splines_4.4.1          later_1.2.0
##   [4] R.oo_1.26.0            cellranger_1.1.0       polyclip_1.10-0
##   [7] XML_3.99-0.17          lifecycle_1.0.4        vroom_1.6.5
##  [10] globals_0.14.0         lattice_0.22-5         MASS_7.3-61
##  [13] backports_1.5.0        magrittr_2.0.3         plotly_4.10.4
##  [16] rmarkdown_2.28         yaml_2.2.1             httpuv_1.6.1
##  [19] sctransform_0.4.1      zip_2.3.1              sp_2.1-4
##  [22] spatstat.sparse_3.1-0  cowplot_1.1.1          pbapply_1.4-3
##  [25] DBI_1.2.3              RColorBrewer_1.1-3     lubridate_1.9.3
##  [28] abind_1.4-5            rvest_1.0.4            Rtsne_0.15
##  [31] purrr_1.0.2            R.utils_2.12.3         rappdirs_0.3.3
##  [34] ggrepel_0.9.6          irlba_2.3.5.1          listenv_0.8.0
##  [37] spatstat.utils_3.1-0   goftest_1.2-2          spatstat.random_3.3-2
##  [40] fitdistrplus_1.1-5     parallelly_1.26.0      leiden_0.3.8
##  [43] codetools_0.2-19       xml2_1.3.6             tidyselect_1.2.1
##  [46] farver_2.1.2           matrixStats_1.4.1      spatstat.explore_3.3-2
##  [49] jsonlite_1.8.9         progressr_0.14.0       ggridges_0.5.3
##  [52] survival_3.7-0         tools_4.4.1            progress_1.2.3
##  [55] ica_1.0-2              Rcpp_1.0.13            glue_1.7.0
##  [58] gridExtra_2.3          xfun_0.47              withr_3.0.1
##  [61] formatR_1.14           fastmap_1.2.0          fansi_1.0.6
##  [64] digest_0.6.37          timechange_0.3.0       R6_2.5.1
##  [67] mime_0.11              colorspace_2.1-1       scattermore_0.7
##  [70] tensor_1.5             spatstat.data_3.1-2    RSQLite_2.3.7
##  [73] R.methodsS3_1.8.2      utf8_1.2.4             generics_0.1.0
##  [76] data.table_1.14.0      prettyunits_1.2.0      httr_1.4.7
##  [79] htmlwidgets_1.5.3      uwot_0.2.2             pkgconfig_2.0.3
##  [82] gtable_0.3.5           blob_1.2.4             lmtest_0.9-38
##  [85] selectr_0.4-2          htmltools_0.5.8.1      png_0.1-8
##  [88] spatstat.univar_3.0-1  knitr_1.48             rstudioapi_0.16.0
##  [91] tzdb_0.4.0             reshape2_1.4.4         checkmate_2.3.2
##  [94] nlme_3.1-165           curl_5.2.3             zoo_1.8-9
##  [97] cachem_1.1.0           stringr_1.5.1          KernSmooth_2.23-24
## [100] parallel_4.4.1         miniUI_0.1.1.1         vipor_0.4.7
## [103] pillar_1.9.0           grid_4.4.1             logger_0.4.0
## [106] vctrs_0.6.5            RANN_2.6.1             promises_1.2.0.1
## [109] xtable_1.8-4           cluster_2.1.6          beeswarm_0.4.0
## [112] evaluate_1.0.0         readr_2.1.5            cli_3.6.3
## [115] compiler_4.4.1         rlang_1.1.4            crayon_1.4.1
## [118] future.apply_1.7.0     labeling_0.4.3         plyr_1.8.6
## [121] ggbeeswarm_0.7.2       stringi_1.6.2          viridisLite_0.4.2
## [124] deldir_2.0-4           BiocParallel_1.38.0    munsell_0.5.1
## [127] lazyeval_0.2.2         spatstat.geom_3.3-3    Matrix_1.6-1.1
```

```
## [130] hms_1.1.3             bit64_4.5.2           future_1.21.0
## [133] shiny_1.9.1           highr_0.11            ROCR_1.0-11
## [136] igraph_1.2.6          memoise_2.0.1         bit_4.5.0
## [139] readxl_1.4.3
```