

Mafb-restricted local monocyte proliferation precedes lung interstitial macrophage differentiation

4-Compare Refilled IMs in Day4 depletion to control

2022-03-09 09:16:14 +0100

Abstract

Resident tissue macrophages (RTM) are differentiated immune cells populating distinct niches and exhibiting important tissue-supportive functions. RTM maintenance is thought to rely on either monocyte engraftment and differentiation, or RTM self-renewal. Here, we developed an inducible mouse model of lung interstitial macrophage (IM) niche depletion and repopulation to investigate IM development *in vivo*. Using time-course single-cell RNA-sequencing analyses, bone marrow chimeras and gene targeting, we found that engrafted Ly6C+ classical monocytes could self-renew locally in a CSF1R-dependent manner before their differentiation into RTM. We further showed that the switch from monocyte proliferation towards IM subset specification was controlled by MafB, while c-Maf specifically regulated the identity of the CD206+ IM subset. Our data shed new light on the transcriptional regulation of IM development and provide evidence that, in the mononuclear phagocyte system, self-renewal is not merely restricted to myeloid progenitor cells and mature macrophages, but is also a tightly regulated capability of mature monocytes developing into RTM *in vivo*.

Contents

| | |
|---|-----------|
| 1 Description | 2 |
| 2 Load data and packages | 2 |
| 3 Build up sample metadata | 2 |
| 4 Celltype identification | 4 |
| 4.1 Clustering cells | 4 |
| 4.2 Identify cells with SingleR | 6 |
| 4.3 Identify cells with marker expression | 6 |
| 4.4 Combine SingleR and marker expression results | 7 |
| 5 Remove other cell types than monocytes/macrophages | 11 |
| 6 Phenotyping of lung monocytes/IMs | 11 |
| 6.1 Clean data after filtering | 11 |
| 6.2 Data re-processing | 11 |
| 6.3 Population characterization | 16 |
| 6.4 Cell-cycle analysis | 18 |
| 7 Session information | 21 |
| References | 24 |

1 Description

We established an in vivo inducible IM-depletion model by generating the IM-DTR mouse line. The IM in these mice expressed specifically the diphtheria toxin receptor (DTR) and 50 ng diphtheria toxin (DT) treatment was showed to be effective to deplete the lung IM population after 24 hours. Then IM population was regenerated from D3. On D7 post-treatment the IM population could recover to a similar size of intact lung IM population.

In this analysis, the lung IM from D4 post-treatment were compared to the intact IM of littermate control mice without DT treatment using 10X single-cell RNAsequencing.

Data processing, cell clustering, linear- or non-linear dimension reduction were made in Seurat package¹ and the celltyping was made by SingleR pakcage.² Cells were firstly clustered with high resolution in order that the contaminated cells would be clustered together and identified with SingleR package. Once the contaminated cells removed, data of the filtered cells were then re-filtered and clustered for further analyses.

2 Load data and packages

```
library(Seurat)
source("../R/seurat.make.integrated.R")  
  
initiation.analysis.folder <- "../3-scRNAseq_initiation_the_IMs_in_Day4_  
depletion/"
file.names <- list.files(initiation.analysis.folder, pattern = "*.rds")
sample.names <- sub(".rds", "", file.names)  
  
table.samples <- data.frame(sample.names, file.names)
i <- sapply(table.samples, is.factor)
table.samples[i] <- lapply(table.samples[i], as.character) # this is for
convert the factor to character.
rm("file.names", "sample.names") # remove individual vector to avoid
confusion.
for (i in 1:length(table.samples$sample.names)) {
  assign(table.samples$sample.names[i], readRDS(file = file.path(
    initiation.analysis.folder, table.samples$file.names[i])))
}
```

3 Build up sample metadata

Here's sample order:

```
table.samples$sample.names  
  
## [1] "CPlus_NGS20_Q147.seuratObject"      "Plusplus_NGS20_Q148.  
seuratObject"
```

Make metadata table

```
# sample.names
sample.metadata <- data.frame(  
  1  
  2
```

```

group = c("Cre+", "++"),
cell.type = rep("lung IM niche (including CD11c)", length(table.samples$sample.names)), # nolint
origin = rep("lung", length(table.samples$sample.names)),
sample_prefix = gsub("_NGS[a-zA-Z0-9_.-]*", "", table.samples$sample.names)
rownames(sample.metadata) <- table.samples$sample.names
sample.metadata

```

| | ## # A tibble: 2 x 4 | origin | sample_prefix |
|---|---|--------|---------------|
| | ## group cell.type | <chr> | <chr> |
| | ## <chr> <chr> | <chr> | <chr> |
| 1 | ## 1 Cre+ lung IM niche (including CD11c) | lung | CPlus |
| 2 | ## 2 ++ lung IM niche (including CD11c) | lung | Plusplus |

```

# add sample information:

for (i in table.samples$sample.names) {
  obj <- get(i)
  obj$treatment <- sample.metadata[i, ]$group
  assign(i, obj)
}

for (i in table.samples$sample.names) {
  obj <- get(i)
  obj$therapy <- sample.metadata[i, ]$cell.type
  assign(i, obj)
}

for (i in table.samples$sample.names) {
  obj <- get(i)
  obj$therapy <- sample.metadata[i, ]$origin
  assign(i, obj)
}

# add pre-fix to cellnames:

for (i in table.samples$sample.names) {
  obj <- get(i)
  obj <- RenameCells(obj,
    add.cell.id = sample.metadata[i, ]$sample_prefix)
  assign(i, obj)
}

```

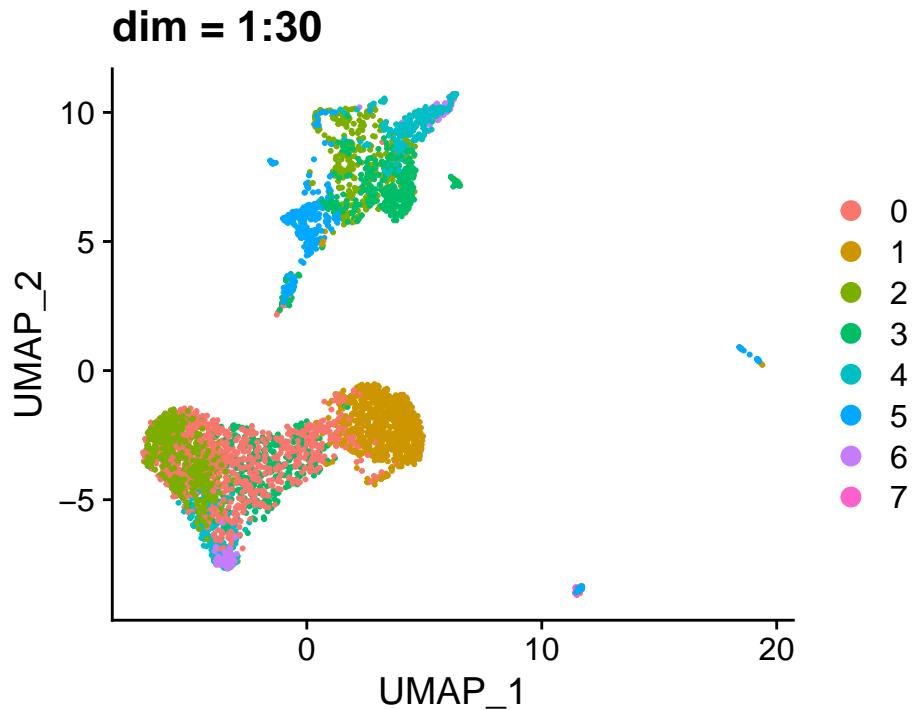
Integrate:

```

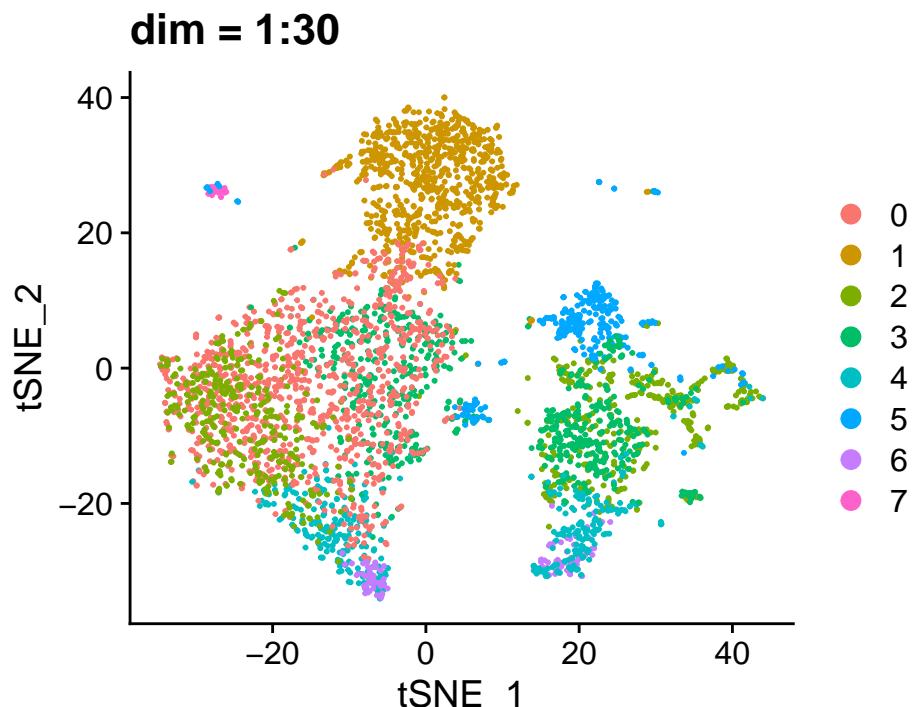
results.dim30 <- seurat.make.integrated(
  seuratObjects = sapply(table.samples$sample.names, get),
  prefix = table.samples$sample.names, dimensionality = 1:30,
  SCTransf = FALSE)

```

```
results.dim30$plots$plot.umap + ggtitle("dim=1:30")
```



```
results.dim30$plots$plot.tsne + ggtitle("dim=1:30")
```



```
results <- results.dim30$integrated.seuratObject
```

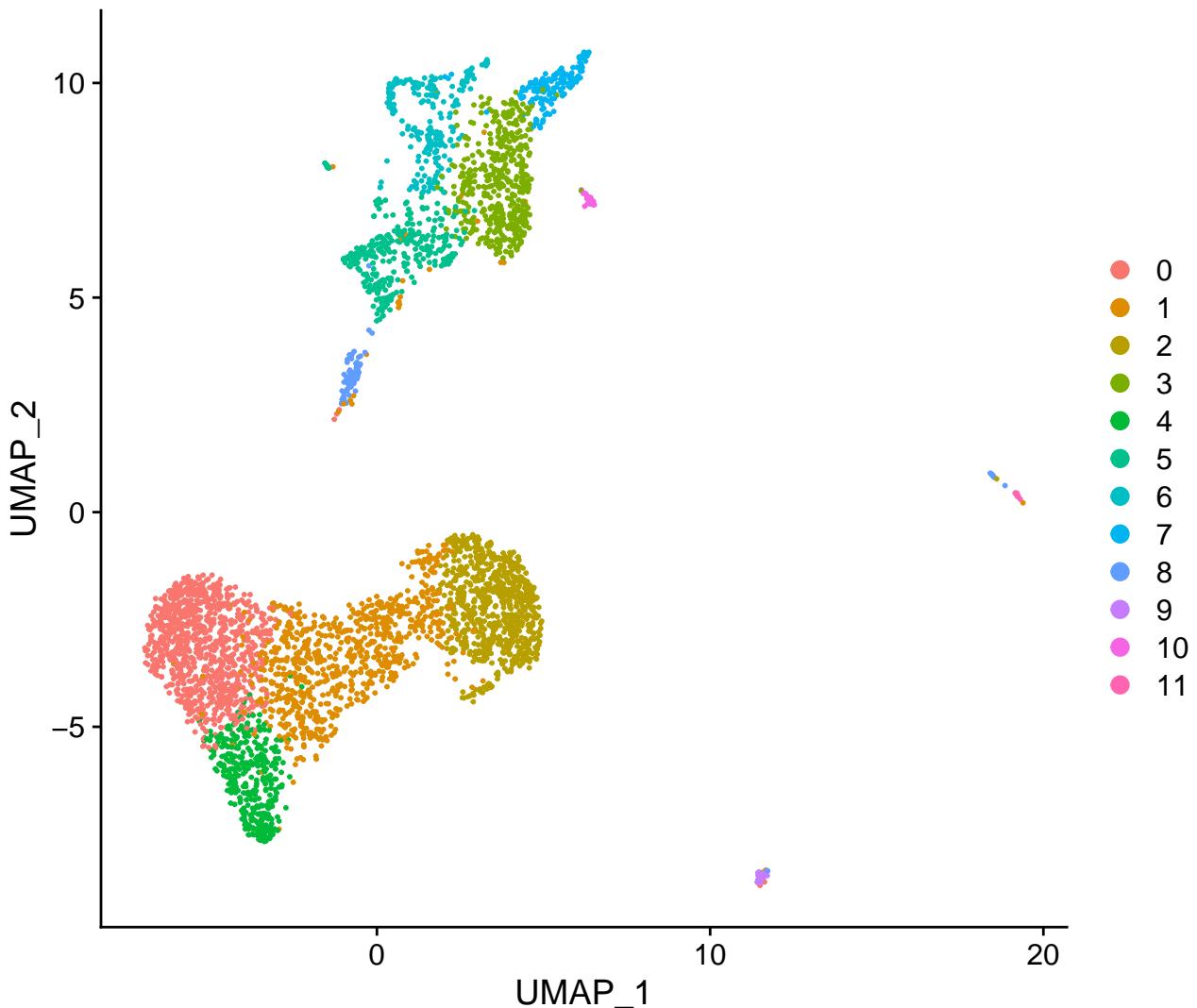
4 Celltype identification

4.1 Clustering cells

```
results <- FindClusters(results, resolution = 0.5)
```

```
## Modularity Optimizer version 1.3.0 by Ludo Waltman and Nees Jan van Eck  
##  
## Number of nodes: 4169  
## Number of edges: 161254  
##  
## Running Louvain algorithm...  
## Maximum modularity in 10 random starts: 0.8723  
## Number of communities: 12  
## Elapsed time: 0 seconds
```

```
DimPlot(results)
```



```
# save data  
saveRDS(results, file = "./results.integrated.Rds")
```

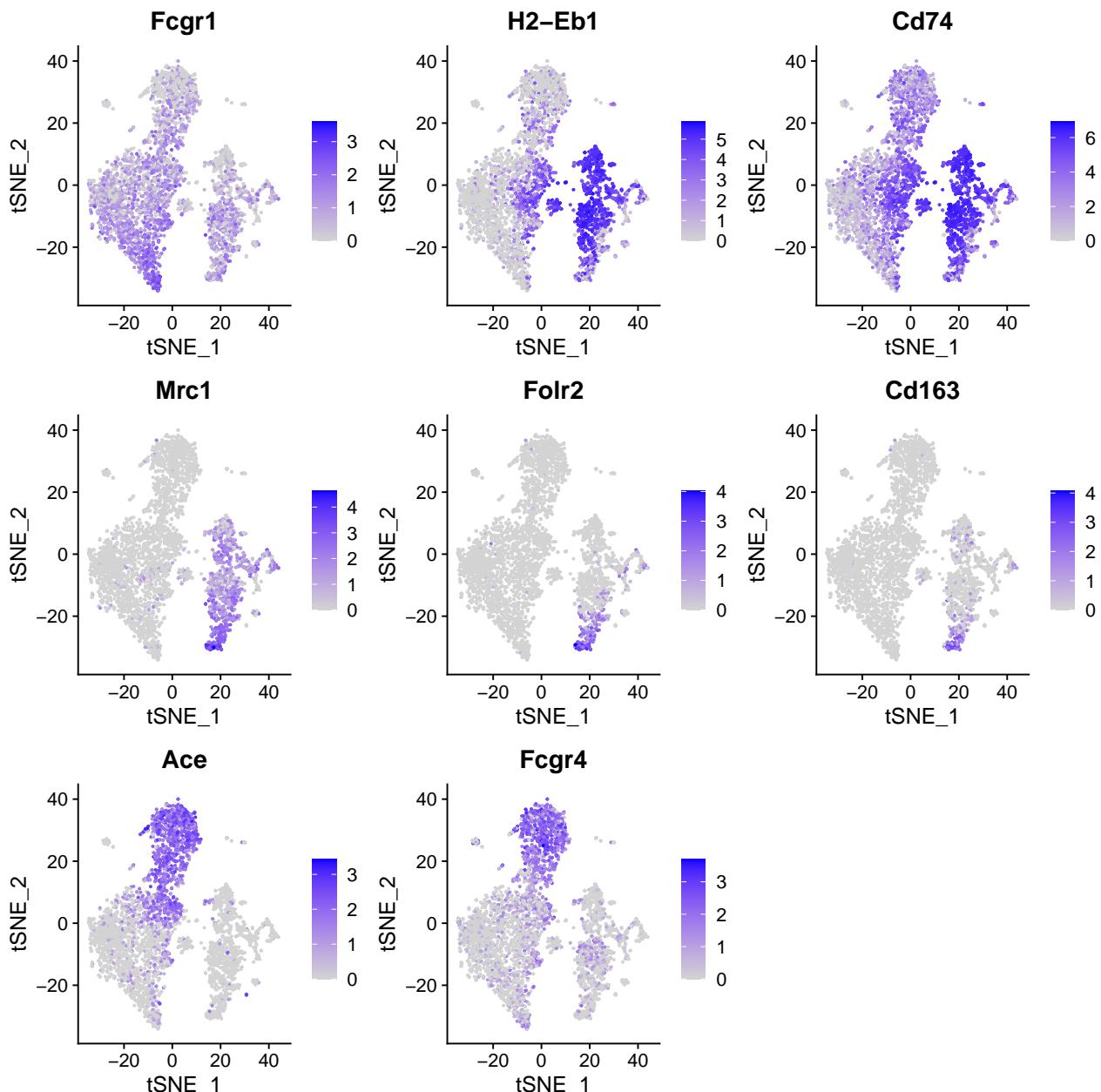
4.2 Identify cells with SingleR

```
source("../R/seurat2singleR.R")
Idents(results) <- "integrated_snn_res.0.5"
results.singleR <- seurat2singleR(results, ref = "ImmGenData")
```

```
saveRDS(results.singleR, file = "./results.ImmGenData.singleR.Rds")
```

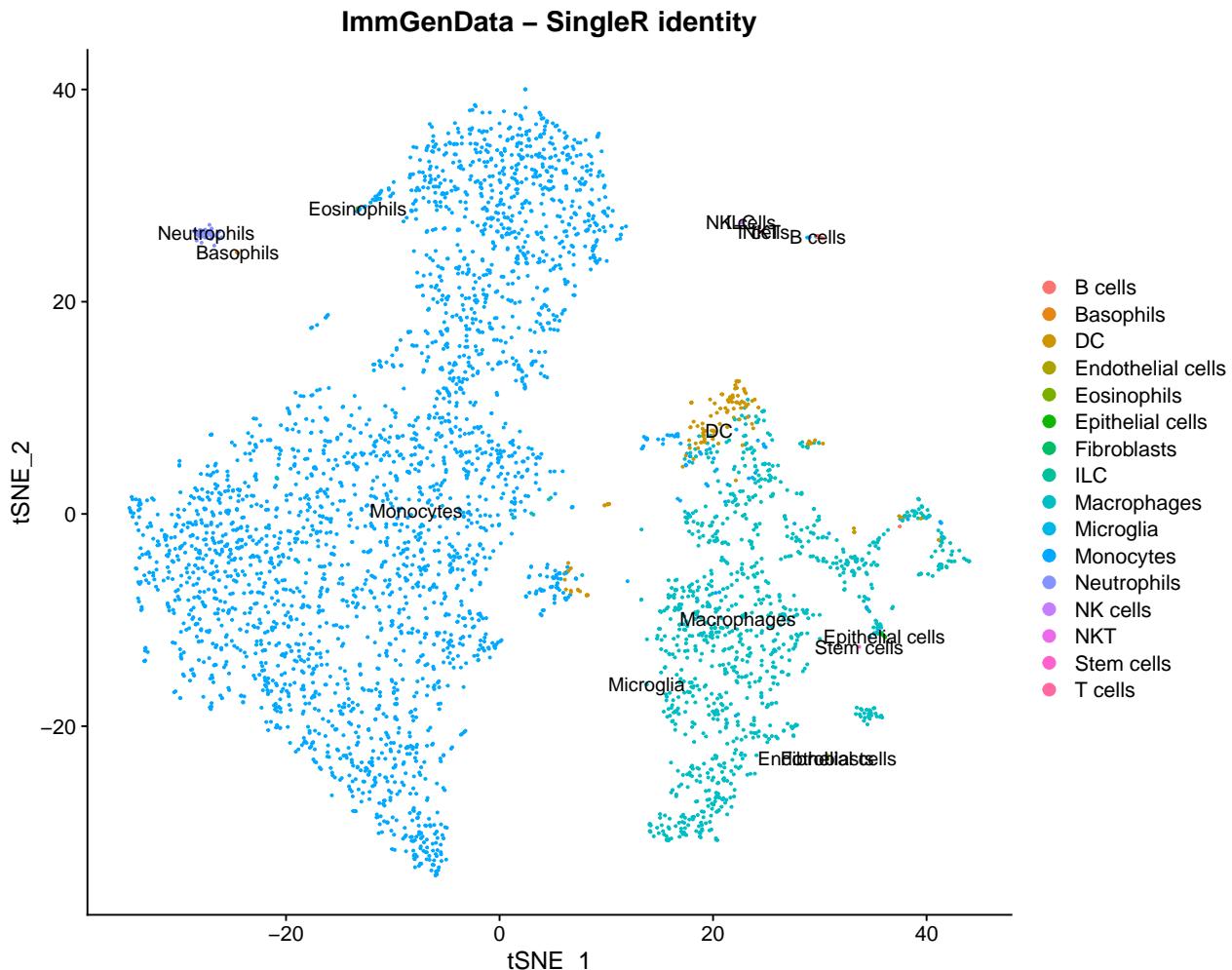
4.3 Identify cells with marker expression

```
DefaultAssay(results) <- "RNA"
FeaturePlot(results, features = c("Fcgr1", "H2-Eb1", "Cd74", "Mrc1", "Folr2", "Cd163", "Ace", "Fcgr4"), reduction = "tsne")
```



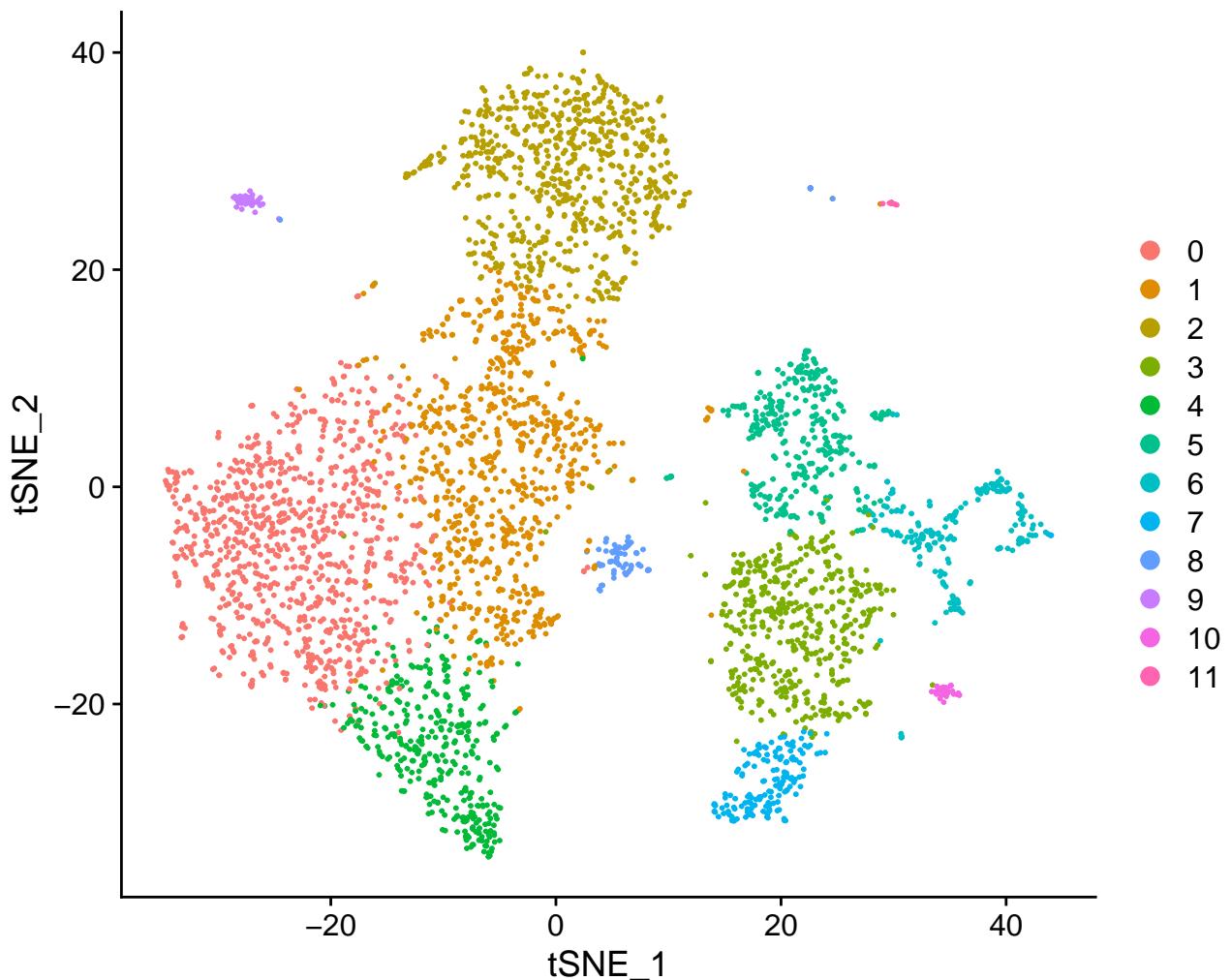
4.4 Combine SingleR and marker expression results

```
results$singleR.celltype <- results.singleR$labels  
DimPlot(results, group.by = "singleR.celltype", reduction = "tsne", label  
= T) + ggtitle("ImmGenData - SingleR identity")  
1  
2
```

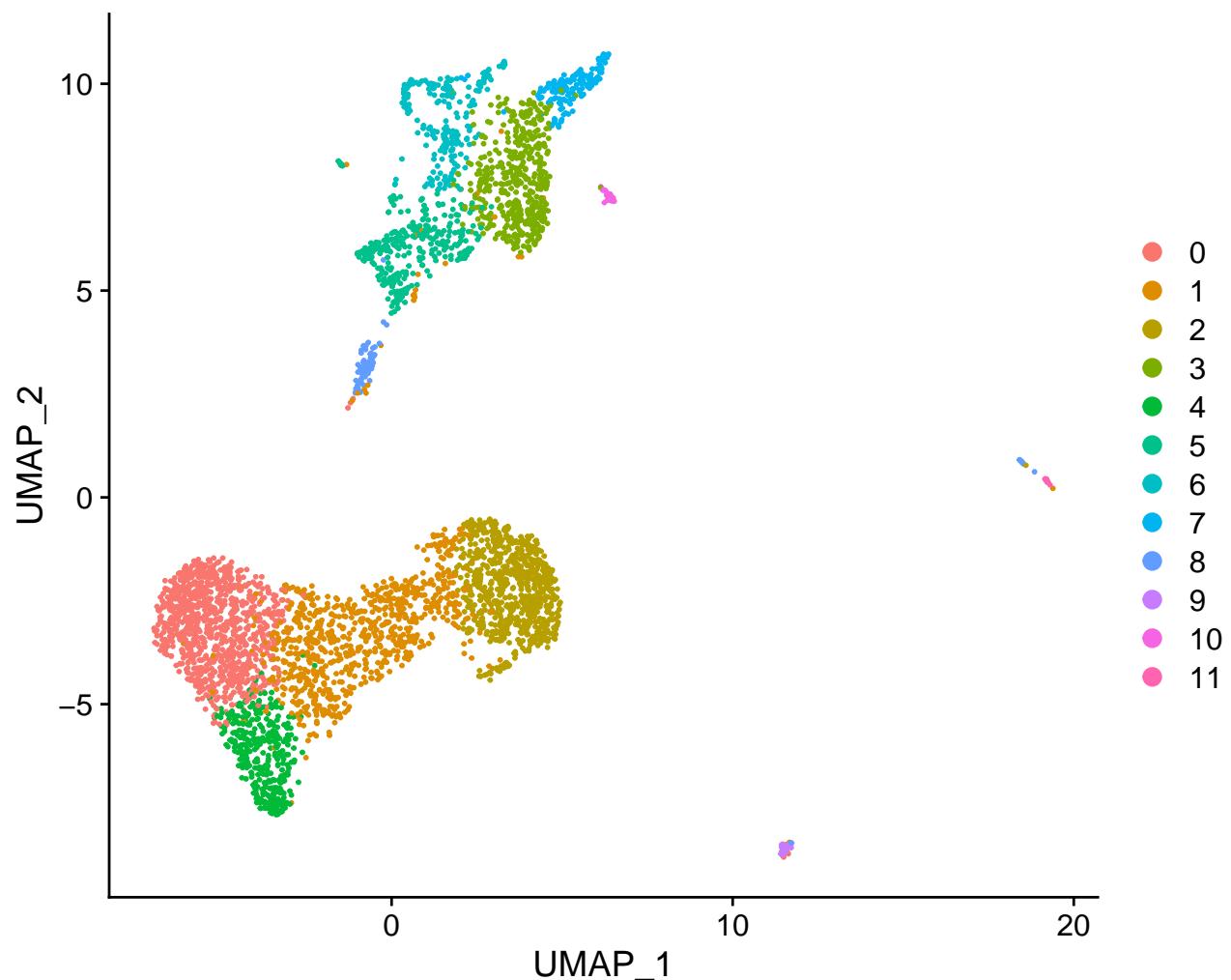


```
DimPlot(results, reduction = "tsne") + ggtitle("results - Clusters")  
1
```

results – Clusters

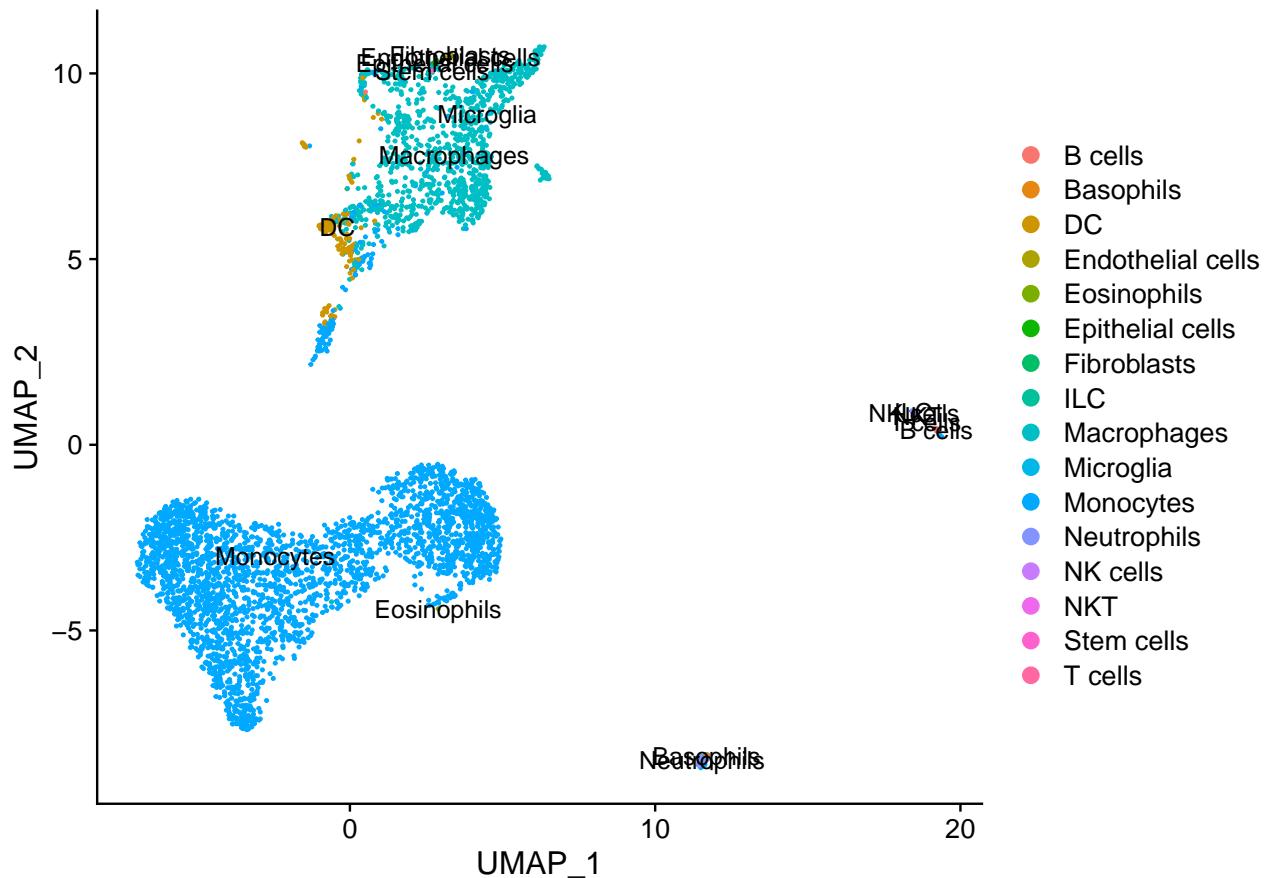


results – Clusters

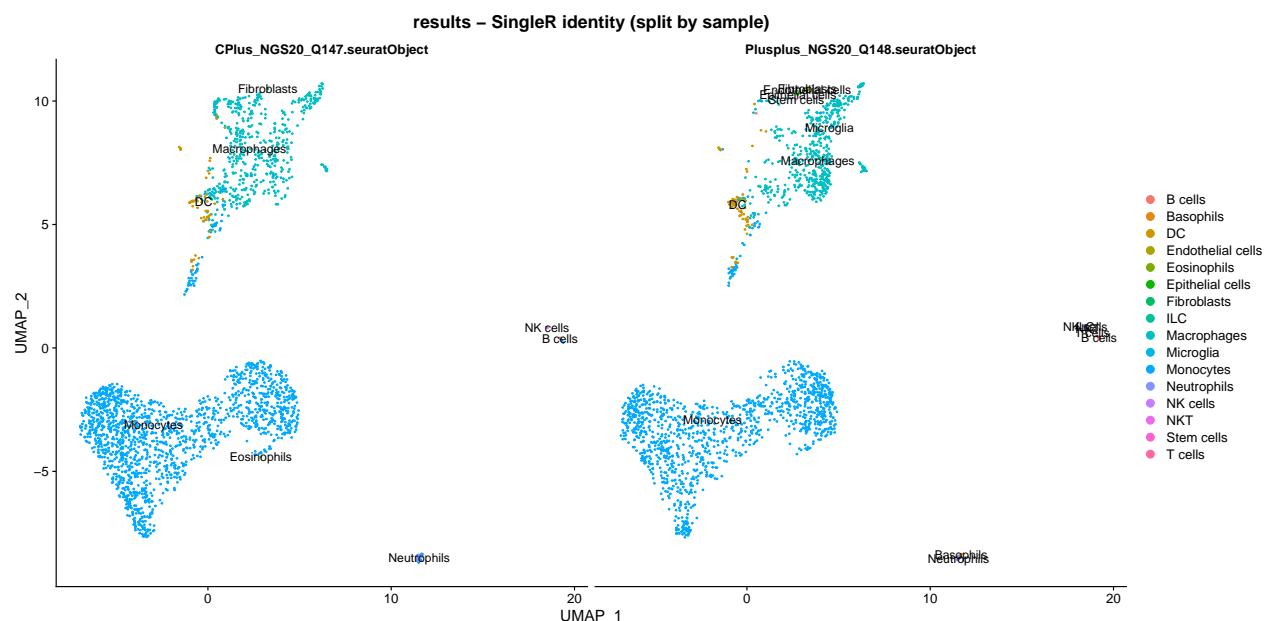


```
DimPlot(results, group.by = "singleR.celltype", reduction = "umap", label = T) + ggttitle("results - SingleR identity")
```

results – SingleR identity



```
DimPlot(results, group.by = "singleR.celltype", reduction = "umap", label = T, split.by = "object_before_integrated", ncol = 2) + ggtitle("results - SingleR identity (split by sample)")
```



5 Remove other cell types than monocytes/macrophages

Cell numbers per cell type:

| | |
|----------------------------------|---|
| table(results\$singleR.celltype) | 1 |
| ## | 1 |
| ## B cells | 2 |
| ## 16 | 3 |
| ## Eosinophils | 2 |
| ## 1 | 3 |
| ## Macrophages | 4 |
| ## 1034 | 5 |
| ## NK cells | 6 |
| ## 7 | 7 |
| Basophils | 8 |
| Epithelial cells | 9 |
| Fibroblasts | 1 |
| Microglia | 2 |
| Monocytes | 3 |
| NKT | 4 |
| Stem cells | 5 |
| T cells | 6 |
| | 7 |
| | 8 |
| | 9 |
| DC | |
| Endothelial cells | |
| ILC | |
| Neutrophils | |
| T cells | |

Keep only “Monocytes”, “Macrophages” and “Microglia”. As microglia are highly similar to lung IMs and we can exclude the possibility of contamination of brain cells in this experiment, we should keep them for the following analysis.

```
results <- results[, WhichCells(results, expression = singleR.celltype %in%
  % c("Monocytes", "Macrophages", "Microglia"))]
```

6 Phenotyping of lung monocytes/IMs

6.1 Clean data after filtering

Remove old snn:

```
results$RNA_snn_res.0.5 <- NULL
results$integrated_snn_res.0.5 <- NULL
```

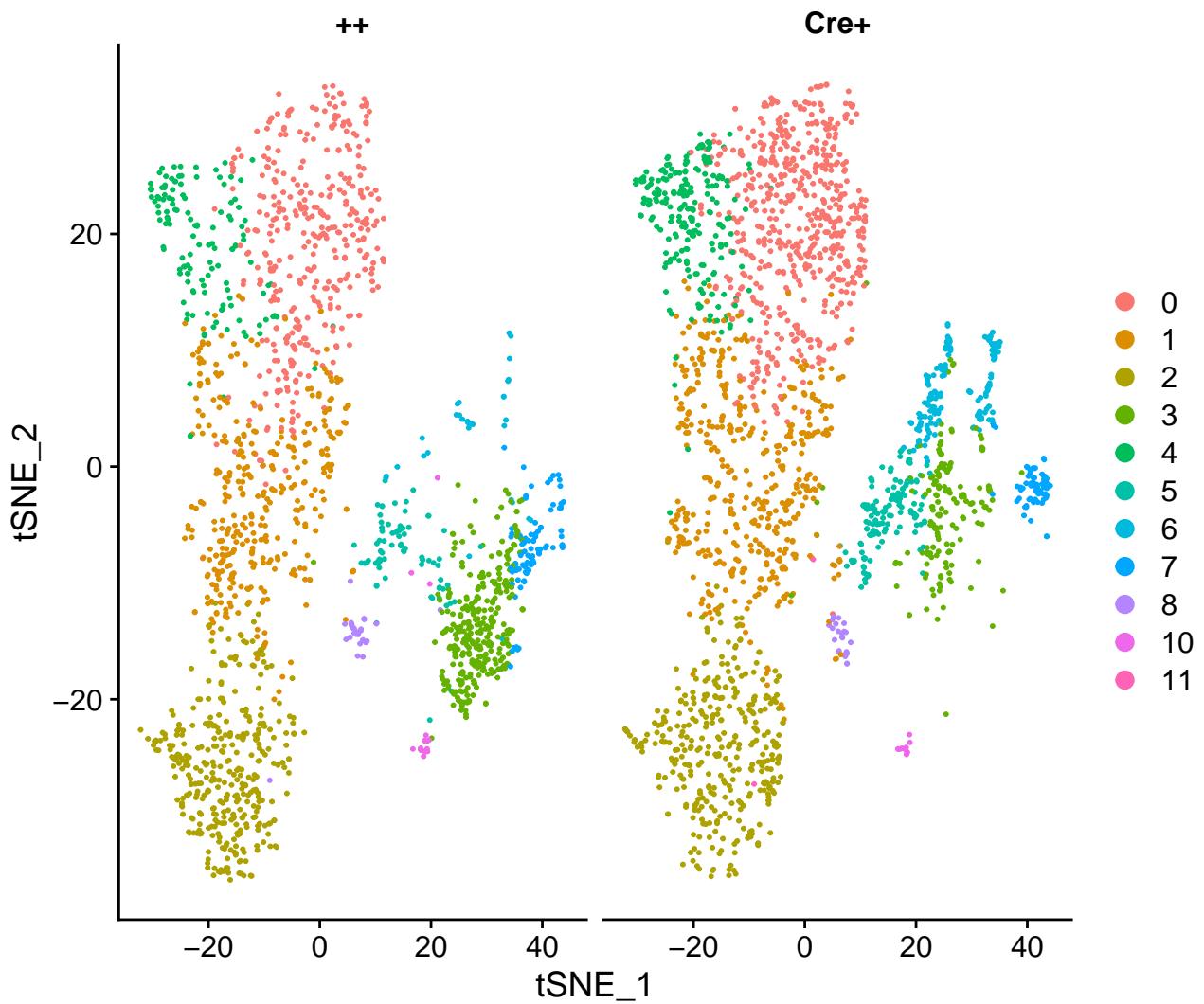
Save data

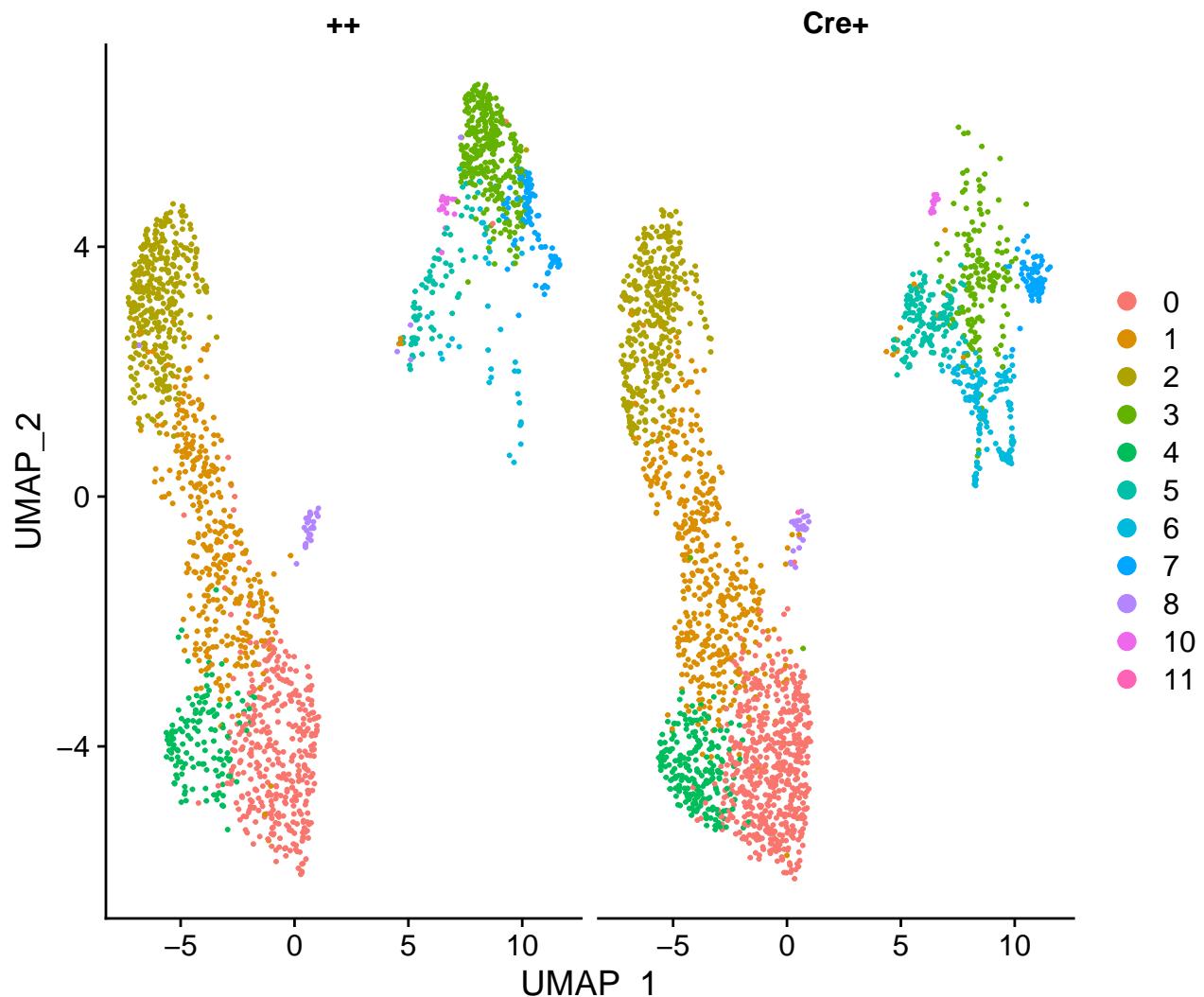
```
saveRDS(results, file = "./results.cellType_filtered.seuratObject.Rds")
```

6.2 Data re-processing

```
DefaultAssay(results) <- "RNA"
results <- NormalizeData(results)
results <- FindVariableFeatures(results, selection.method = "vst",
  nfeatures = 2000)
results <- ScaleData(results, features = rownames(results))
results <- RunPCA(results, features = VariableFeatures(results))
results <- RunTSNE(results, dims = 1:20)
results <- RunUMAP(results, dims = 1:20)

DimPlot(results, reduction = "tsne", split.by = "treatment")
```





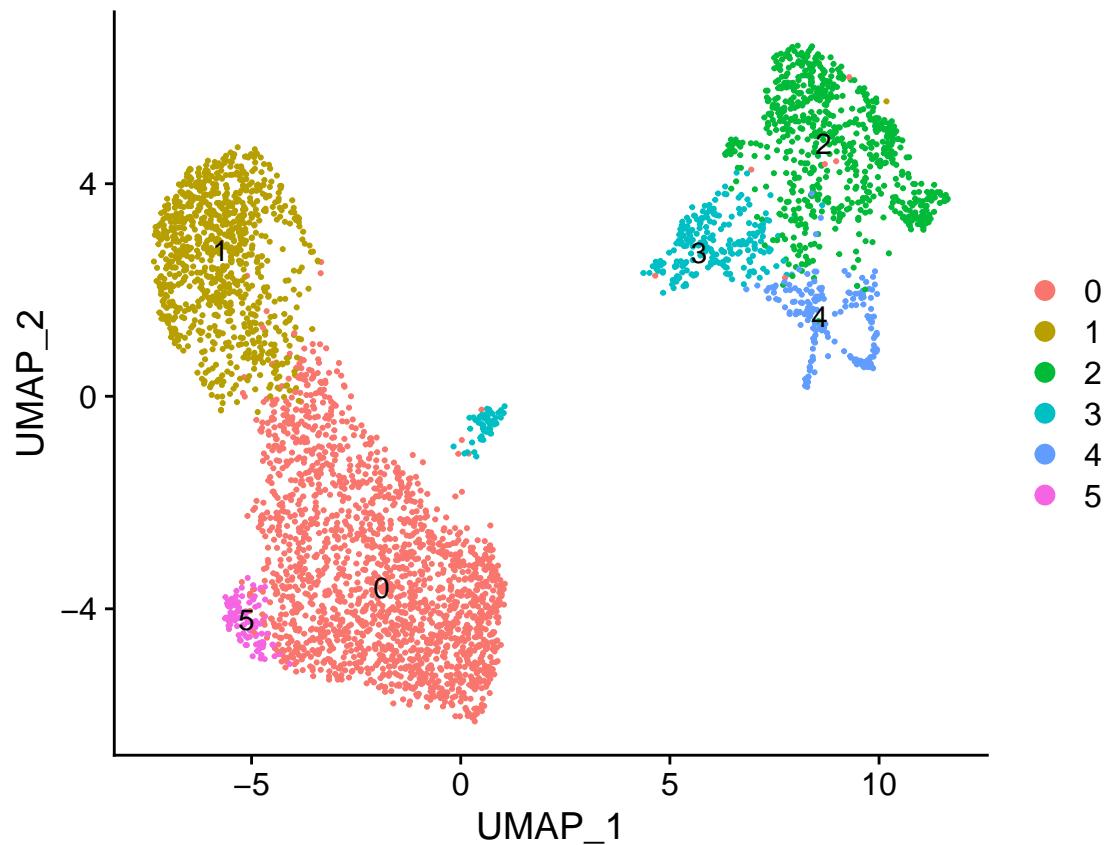
Clustering of cells

```
results <- FindNeighbors(results, dims = 1:20)
results <- FindClusters(results, resolution = 0.3)
```

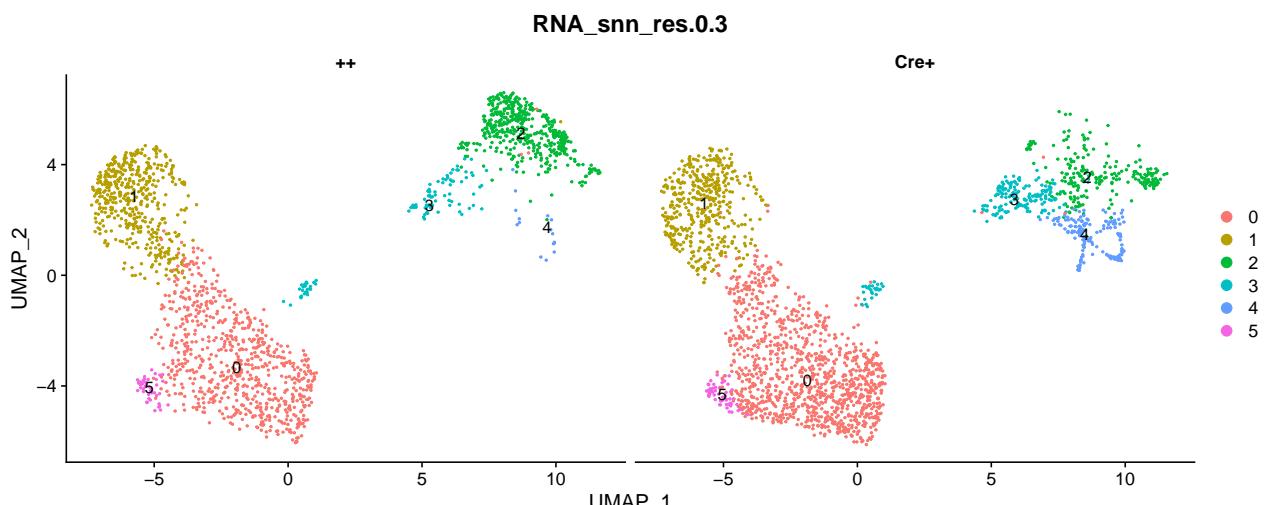
```
## Modularity Optimizer version 1.3.0 by Ludo Waltman and Nees Jan van Eck
## Number of nodes: 3958
## Number of edges: 139307
## Running Louvain algorithm...
## Maximum modularity in 10 random starts: 0.8872
## Number of communities: 6
## Elapsed time: 0 seconds
```

```
Idents(results) <- "RNA_snn_res.0.3"
DimPlot(results, group.by = "RNA_snn_res.0.3", label = TRUE)
```

RNA_snn_res.0.3

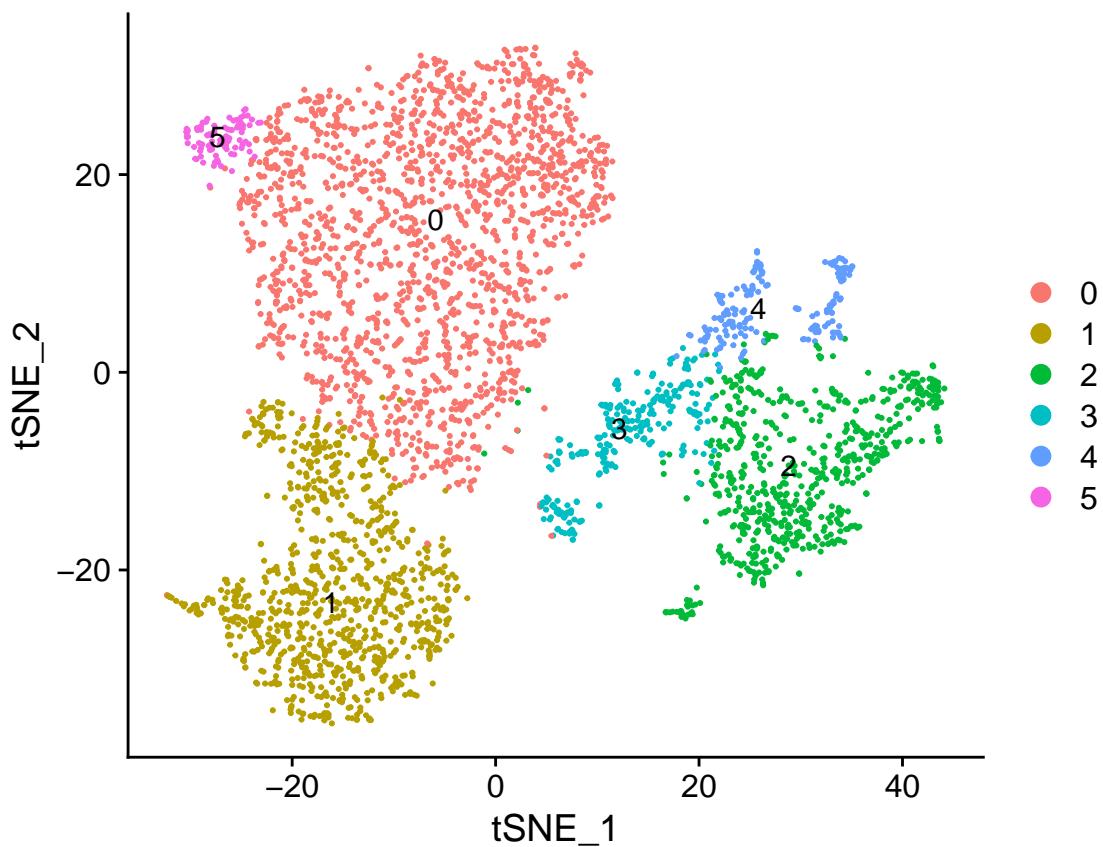


```
DimPlot(results, split.by = "treatment", group.by = "RNA_snn_res.0.3",  
        label = TRUE)
```

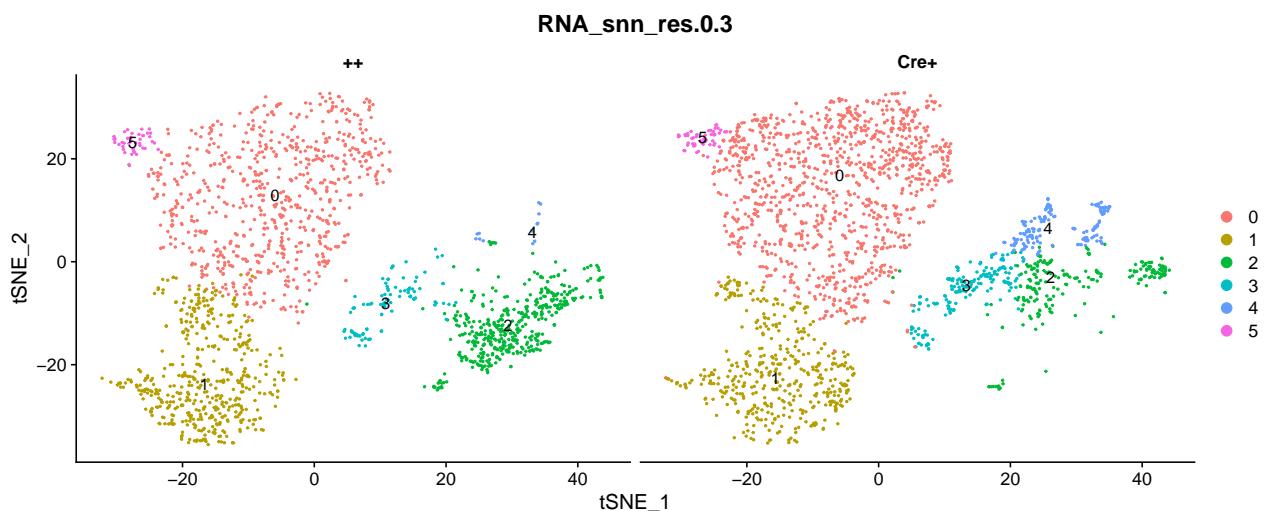


```
DimPlot(results, group.by = "RNA_snn_res.0.3", reduction = "tsne", label = TRUE)
```

RNA_snn_res.0.3



```
DimPlot(results, split.by = "treatment", group.by = "RNA_snn_res.0.3",
        reduction = "tsne", label = TRUE)
```



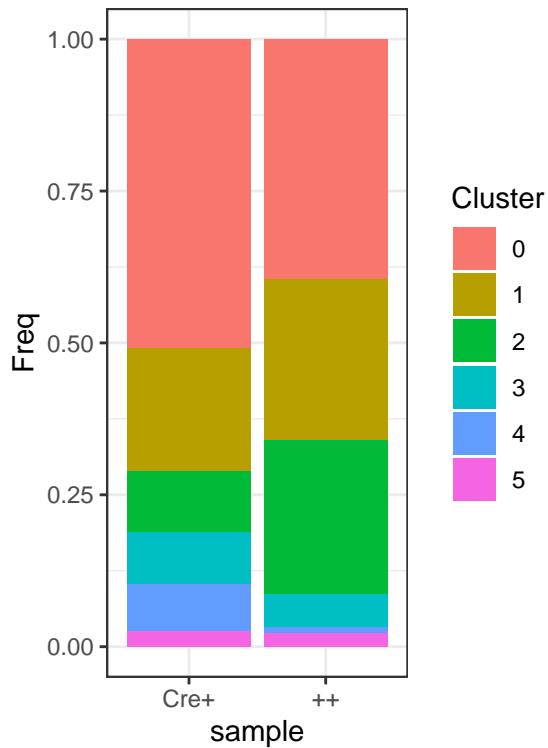
Distribution of each cluster across the samples

```
source("../R/SeuratFreqTable.R")
freq.celltype.list <- list(
  `Cre+` = Seurat2CellFreqTable(subset(results, subset = treatment == "Cre +
  +"), slotName = "RNA_snn_res.0.3"),
```

```

`++` = Seurat2CellFreqTable(subset(results, subset = treatment == "++"),
                           slotName = "RNA_snn_res.0.3")
)

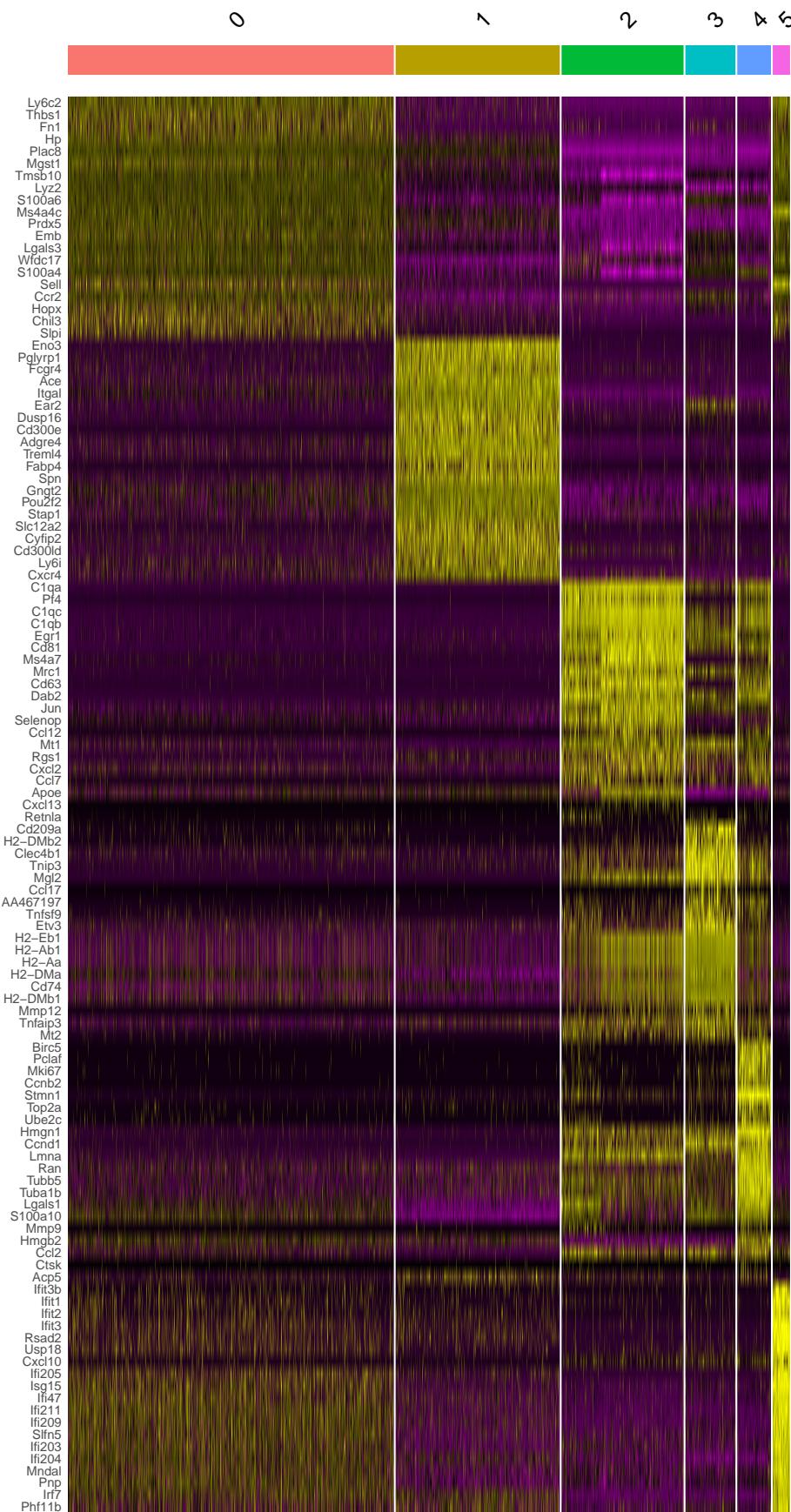
source("../R/barChart.R")
barChart(freq.celltype.list) + labs(fill = "Cluster")
```



6.3 Population characterization

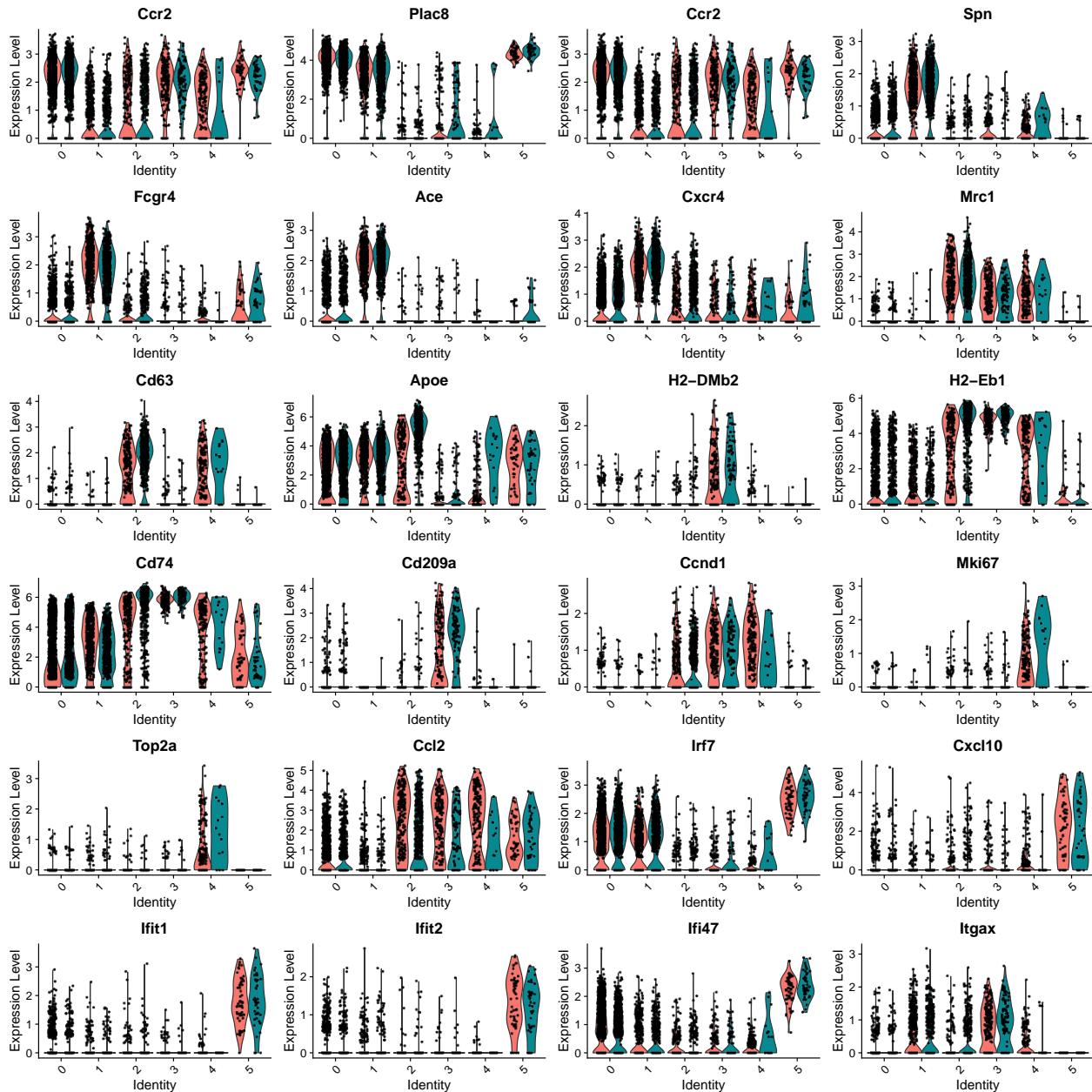
```

library(dplyr)
all_cluster.markers <- FindAllMarkers(results)
top20 <- all_cluster.markers %>% group_by(cluster) %>% top_n(n = 20, wt =
  avg_log2FC)
DoHeatmap(results, features = top20$gene) + NoLegend()
```



Expression of markers:

```
VlnPlot(results, features = c("Ccr2", "Plac8", "Ccr2", "Spn",
                               "Fcgr4", "Ace", "Cxcr4",
                               "Mrc1", "Cd63", "Apoe",
                               "H2-DMb2", "H2-Eb1", "Cd74", "Cd209a",
                               "Ccnd1", "Mki67", "Top2a", "Ccl2",
                               "Irf7", "Cxcl10", "Ifit1", "Ifit2", "Ifi47",
                               "Itgax"),
       split.by = "object_before_integrated", ncol = 4)
```



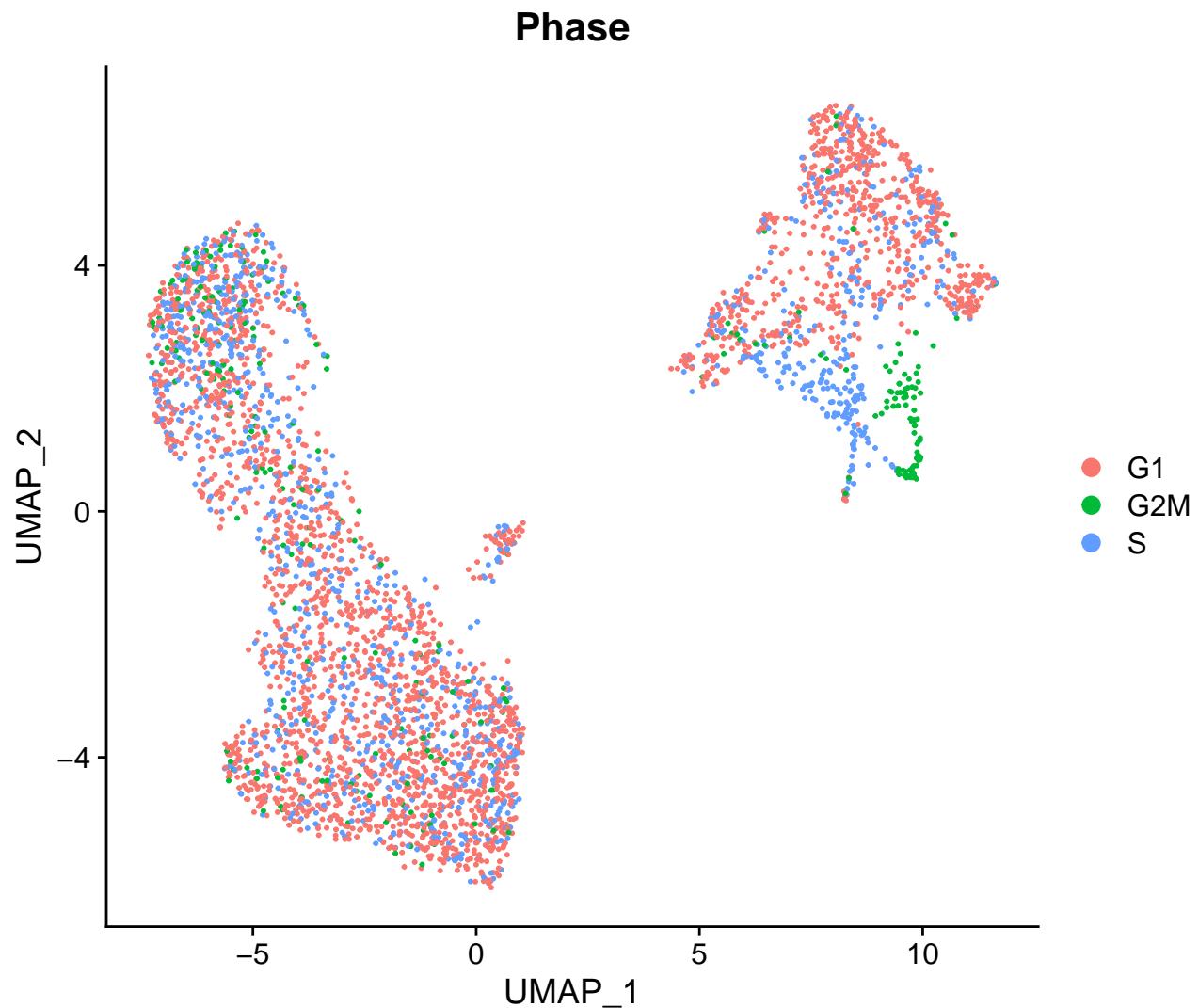
6.4 Cell-cycle analysis

```
library(cowplot)
```

```

1 data("geneinfo_human", package = "nichenetr")
2 s.genes <- nichenetr::convert_human_to_mouse_symbols(cc.genes.updated.2019
3   $s.genes)
4 g2m.genes <- nichenetr::convert_human_to_mouse_symbols(cc.genes.updated
5   .2019$g2m.genes)
6 results <- CellCycleScoring(results, s.features = s.genes, g2m.features =
6   g2m.genes, set.ident = FALSE)
7 DimPlot(results, group.by = "Phase")

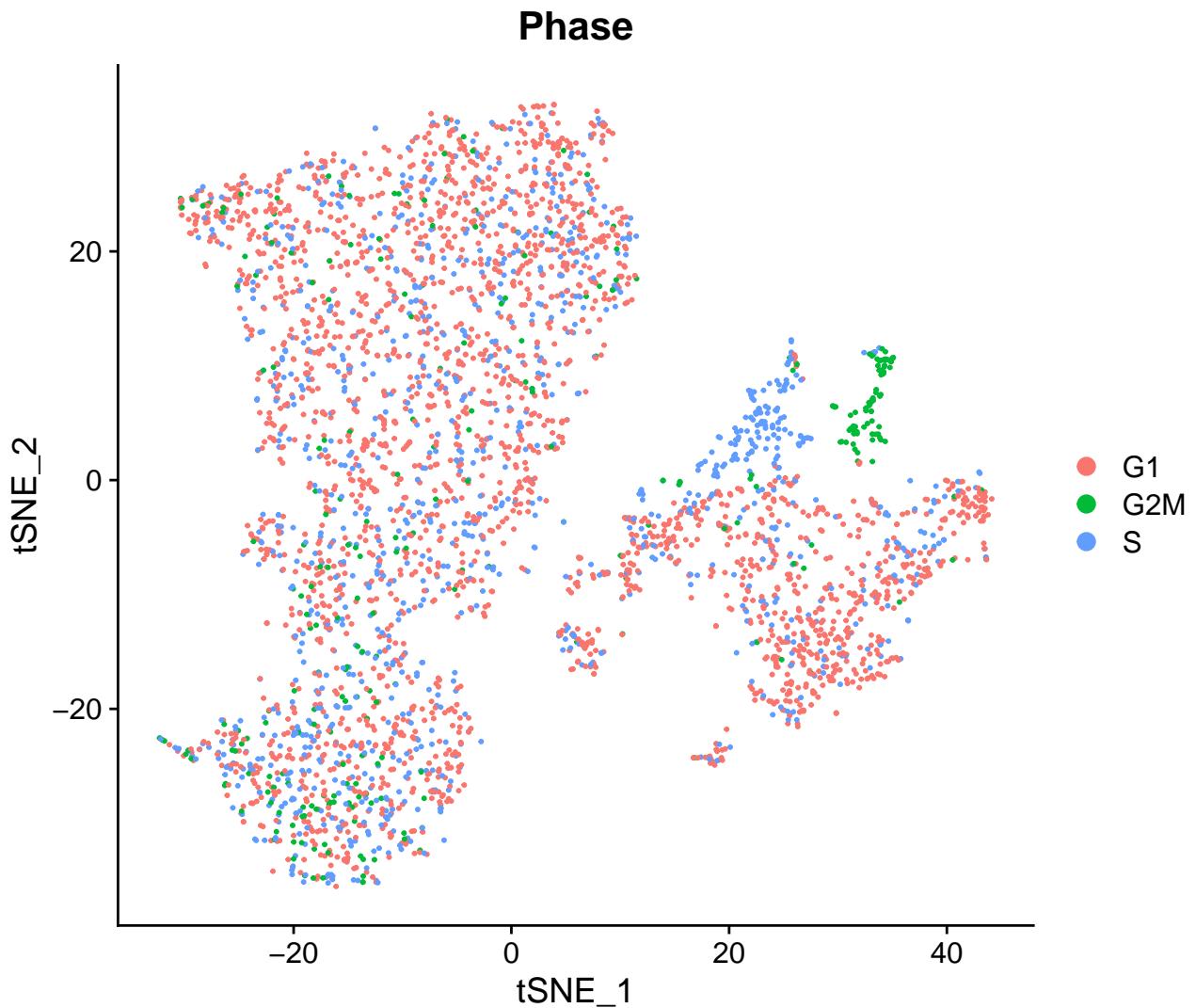
```



```

1 DimPlot(results, group.by = "Phase", reduction = "tsne")

```

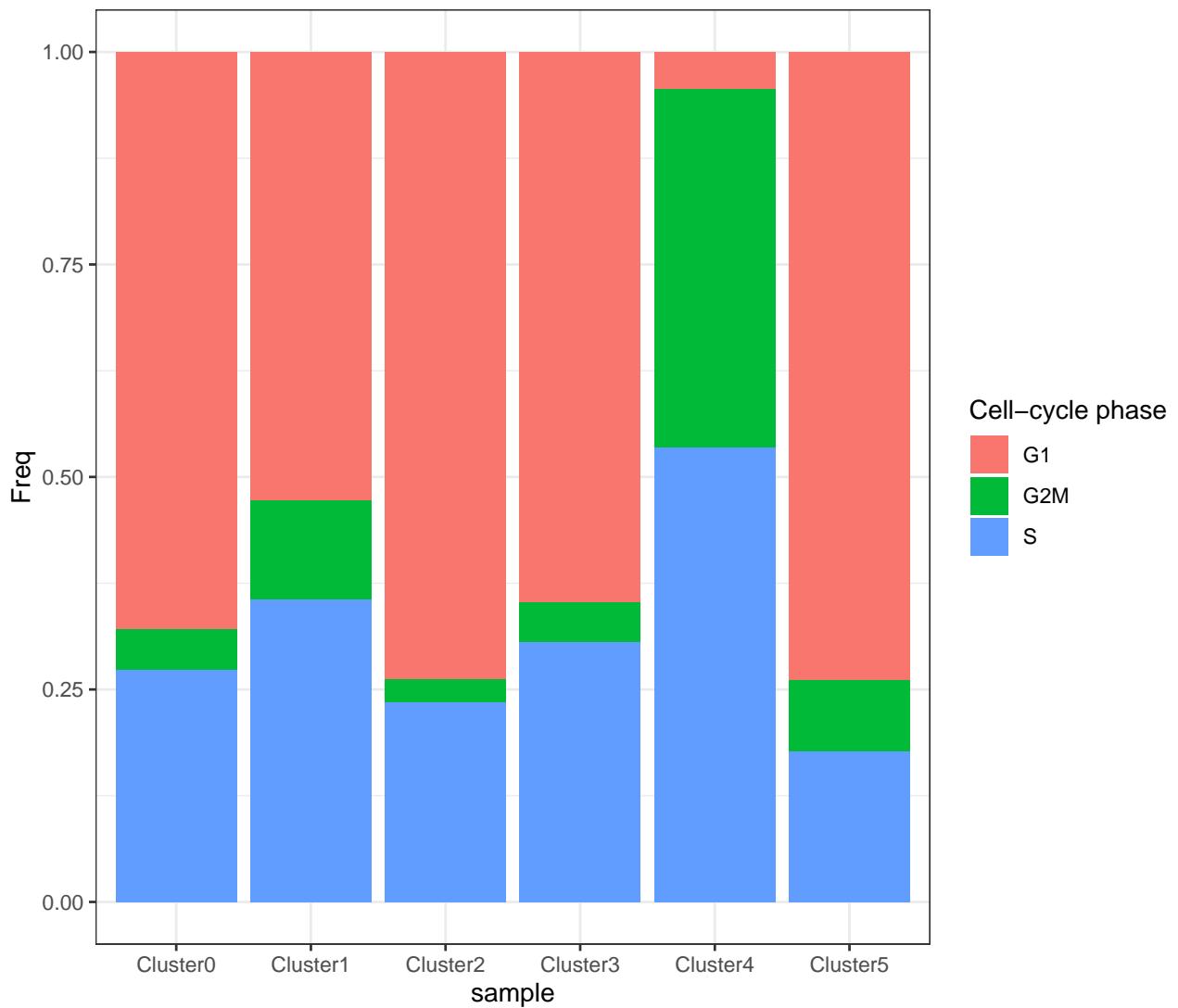


Distribution of cell phases

```

freq.celltype.list <- list(
  Cluster0 = Seurat2CellFreqTable(subset(results, ident = 0), slotName = "1
  Phase"),
  Cluster1 = Seurat2CellFreqTable(subset(results, ident = 1), slotName = "2
  Phase"),
  Cluster2 = Seurat2CellFreqTable(subset(results, ident = 2), slotName = "3
  Phase"),
  Cluster3 = Seurat2CellFreqTable(subset(results, ident = 3), slotName = "4
  Phase"),
  Cluster4 = Seurat2CellFreqTable(subset(results, ident = 4), slotName = "5
  Phase"),
  Cluster5 = Seurat2CellFreqTable(subset(results, ident = 5), slotName = "6
  Phase")
)
barChart(freq.celltype.list) + labs(fill = "Cell-cycle phase")7
8
9

```



7 Session information

| sessionInfo() | |
|--|----|
| ## R version 4.0.3 (2020-10-10) | 1 |
| ## Platform: x86_64-pc-linux-gnu (64-bit) | 2 |
| ## Running under: Ubuntu 20.04.3 LTS | 3 |
| ## | 4 |
| ## Matrix products: default | 5 |
| ## BLAS: /usr/lib/x86_64-linux-gnu/openblas-pthread/libblas.so.3 | 6 |
| ## LAPACK: /usr/lib/x86_64-linux-gnu/openblas-pthread/liblapack.so.3 | 7 |
| ## | 8 |
| ## locale: | 9 |
| ## [1] LC_CTYPE=en_US.UTF-8 LC_NUMERIC=C | 10 |
| ## [3] LC_TIME=en_GB.UTF-8 LC_COLLATE=en_US.UTF-8 | 11 |
| ## [5] LC_MONETARY=en_GB.UTF-8 LC_MESSAGES=en_US.UTF-8 | 12 |
| ## [7] LC_PAPER=en_GB.UTF-8 LC_NAME=C | 13 |
| ## [9] LC_ADDRESS=C LC_TELEPHONE=C | 14 |

```

## [11] LC_MEASUREMENT=en_GB.UTF-8 LC_IDENTIFICATION=C          15
##
## attached base packages:
## [1] parallel    stats      graphics   grDevices utils       16
##           datasets
## [8] methods     base        17
##
## other attached packages:
## [1] cowplot_1.1.1           dplyr_1.0.8            18
## [3] RColorBrewer_1.1-2      celldex_1.0.0          19
## [5] SingleR_1.4.1          SummarizedExperiment_1.20.0 20
## [7] Biobase_2.50.0          GenomicRanges_1.42.0    21
## [9] GenomeInfoDb_1.26.7      IRanges_2.24.1          22
## [11] S4Vectors_0.28.1        BiocGenerics_0.36.1    23
## [13] MatrixGenerics_1.2.1   matrixStats_0.61.0    24
## [15] ggplot2_3.3.5           SeuratObject_4.0.4     25
## [17] Seurat_4.1.0            26
##
## loaded via a namespace (and not attached):
## [1] utf8_1.2.2                reticulate_1.24         27
## [3] tidyselect_1.1.1           RSQLite_2.2.10          28
## [5] AnnotationDbi_1.52.0      htmlwidgets_1.5.4        29
## [7] grid_4.0.3                 BiocParallel_1.24.1     30
## [9] Rtsne_0.15                pROC_1.18.0             31
## [11] munsell_0.5.0             codetools_0.2-18        32
## [13] ica_1.0-2                 future_1.24.0           33
## [15] miniUI_0.1.1.1            withr_2.4.3              34
## [17] spatstat.random_2.1-0      colorspace_2.0-3        35
## [19] highr_0.9                 knitr_1.37              36
## [21] rstudioapi_0.13            ROCR_1.0-11              37
## [23] tensor_1.5                listenv_0.8.0            38
## [25] labeling_0.4.2             GenomeInfoDbData_1.2.4 39
## [27] polyclip_1.10-0            bit64_4.0.5              40
## [29] farver_2.1.0              parallelly_1.30.0        41
## [31] vctrs_0.3.8                generics_0.1.2           42
## [33] ipred_0.9-12              xfun_0.29               43
## [35] BiocFileCache_1.14.0       randomForest_4.6-14      44
## [37] R6_2.5.1                  ggbeeswarm_0.6.0          45
## [39] rsvd_1.0.5                bitops_1.0-7             46
## [41] spatstat.utils_2.3-0       cachem_1.0.6             47
## [43] DelayedArray_0.16.3       assertthat_0.2.1         48
## [45] promises_1.2.0.1           scales_1.1.1              49
## [47] nnet_7.3-14               beeswarm_0.4.0            50
## [49] gtable_0.3.0              beachmat_2.6.4            51
## [51] Cairo_1.5-14              globals_0.14.0            52
## [53] goftest_1.2-3              timeDate_3043.102        53
## [55] rlang_1.0.1                splines_4.0.3             54
## [57] lazyeval_0.2.2              ModelMetrics_1.2.2.2      55
## [59] checkmate_2.0.0             spatstat.geom_2.3-2        56
## [61] BiocManager_1.30.16         yaml_2.3.5               57
## [63] reshape2_1.4.4              abind_1.4-5              58
## [65] backports_1.4.1             httpuv_1.6.5             59
## [67] Hmisc_4.6-0                 DiagrammeR_1.0.8          60
## [69] caret_6.0-90               lava_1.6.10              61

```

| | | |
|--|-------------------------|-----|
| ## [71] tools_4.0.3 | ellipsis_0.3.2 | 68 |
| ## [73] spatstat.core_2.4-0 | proxy_0.4-26 | 69 |
| ## [75] ggridges_0.5.3 | Rcpp_1.0.8 | 70 |
| ## [77] plyr_1.8.6 | base64enc_0.1-3 | 71 |
| ## [79] visNetwork_2.1.0 | sparseMatrixStats_1.2.1 | 72 |
| ## [81] zlibbioc_1.36.0 | purrr_0.3.4 | 73 |
| ## [83] RCurl_1.98-1.6 | rpart_4.1-15 | 74 |
| ## [85] deldir_1.0-6 | pbapply_1.5-0 | 75 |
| ## [87] zoo_1.8-9 | nichenetr_1.0.0 | 76 |
| ## [89] ggrepel_0.9.1 | cluster_2.1.0 | 77 |
| ## [91] magrittr_2.0.2 | data.table_1.14.2 | 78 |
| ## [93] RSpecs_0.16-0 | scattermore_0.8 | 79 |
| ## [95] lmtest_0.9-39 | RANN_2.6.1 | 80 |
| ## [97] fitdistrplus_1.1-6 | hms_1.1.1 | 81 |
| ## [99] patchwork_1.1.1 | mime_0.12 | 82 |
| ## [101] evaluate_0.15 | xtable_1.8-4 | 83 |
| ## [103] jpeg_0.1-9 | gridExtra_2.3 | 84 |
| ## [105] compiler_4.0.3 | tibble_3.1.6 | 85 |
| ## [107] KernSmooth_2.23-20 | crayon_1.5.0 | 86 |
| ## [109] htmltools_0.5.2 | tzdb_0.2.0 | 87 |
| ## [111] mgcv_1.8-33 | later_1.3.0 | 88 |
| ## [113] Formula_1.2-4 | tidyR_1.2.0 | 89 |
| ## [115] lubridate_1.8.0 | DBI_1.1.2 | 90 |
| ## [117] ExperimentHub_1.16.1 | dbplyr_2.1.1 | 91 |
| ## [119] MASS_7.3-53 | rappdirs_0.3.3 | 92 |
| ## [121] readr_2.1.2 | Matrix_1.4-0 | 93 |
| ## [123] cli_3.2.0 | gower_1.0.0 | 94 |
| ## [125] igraph_1.2.11 | pkgconfig_2.0.3 | 95 |
| ## [127] foreign_0.8-82 | recipes_0.2.0 | 96 |
| ## [129] plotly_4.10.0 | spatstat.sparse_2.1-0 | 97 |
| ## [131] foreach_1.5.2 | hardhat_0.2.0 | 98 |
| ## [133] vipor_0.4.5 | XVector_0.30.0 | 99 |
| ## [135] prodlim_2019.11.13 | stringr_1.4.0 | 100 |
| ## [137] digest_0.6.29 | sctransform_0.3.3 | 101 |
| ## [139] RcppAnnoy_0.0.19 | spatstat.data_2.1-2 | 102 |
| ## [141] rmarkdown_2.11 | leiden_0.3.9 | 103 |
| ## [143] htmlTable_2.4.0 | uwot_0.1.11 | 104 |
| ## [145] DelayedMatrixStats_1.12.3 | curl_4.3.2 | 105 |
| ## [147] shiny_1.7.1 | lifecycle_1.0.1 | 106 |
| ## [149] nlme_3.1-155 | jsonlite_1.7.3 | 107 |
| ## [151] BiocNeighbors_1.8.2 | viridisLite_0.4.0 | 108 |
| ## [153] limma_3.46.0 | fansi_1.0.2 | 109 |
| ## [155] pillar_1.7.0 | lattice_0.20-41 | 110 |
| ## [157] ggrastr_1.0.1 | fastmap_1.1.0 | 111 |
| ## [159] httr_1.4.2 | survival_3.2-7 | 112 |
| ## [161] interactiveDisplayBase_1.28.0 | glue_1.6.1 | 113 |
| ## [163] fdrtool_1.2.17 | png_0.1-7 | 114 |
| ## [165] iterators_1.0.14 | BiocVersion_3.12.0 | 115 |
| ## [167] bit_4.0.4 | class_7.3-17 | 116 |
| ## [169] stringi_1.7.6 | blob_1.2.2 | 117 |
| ## [171] BiocSingular_1.6.0 | AnnotationHub_2.22.1 | 118 |
| ## [173] caTools_1.18.2 | latticeExtra_0.6-29 | 119 |
| ## [175] memoise_2.0.1 | e1071_1.7-9 | 120 |
| ## [177] irlba_2.3.5 | future.apply_1.8.1 | 121 |

References

1. Hao, Y. *et al.* Integrated analysis of multimodal single-cell data. *Cell* (2021) doi:10.1016/j.cell.2021.04.048.
2. Aran, D. *et al.* Reference-based analysis of lung single-cell sequencing reveals a transitional profibrotic macrophage. *Nature Immunology* **20**, 163–172 (2019).