Mafb-restricted local monocyte proliferation precedes lung interstitial macrophage differentiation

6-Merge two experiments and plot presentation

2022-03-09 14:48:23 +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

To obtain a dynamic expression profile from refilled monocytes to IM differentiation, we analyzed lung monocytes and IM of IM-DTR mice at different time points of diphtheria toxin (DT) treatment.

Here, we combined the monocytes/IM data from two experiements (raw data and processed data could be found under accession number GSE193894 and GSE193896 in NCBI GEO). We illustrated the different cell types in a 3-dimension plot and found a transit population bridging classical monocytes and IM.

2 Load data and packages

```
# packages
library(Seurat)
library(ggplot2)
```

```
# data
imdtr1 <- readRDS("../3-scRNAseq_initiation_the_IMs_in_Day4_depletion/
   results.withSingleR.Rds")
imdtr2 <- readRDS("../5-scRNAseq_initiation_the_IMs_refilling_during_post_
   deletion/immune.cellType_filtered.withSingleR.seuratObject.Rds")
```

3 Merge Exp1 and Exp2

3.1 Prepare metadata:

```
imdtr1$orig.ident <- "IM-DTR1"
imdtr2$orig.ident <- "IM-DTR2"

imdtr1$cell.type0 <- "CD45+"

imdtr1$cell.type2 <- imdtr1$RNA_snn_res.0.3

imdtr1$treatment <- ifelse(imdtr1$treatment=="Cre+", "4d", no = "0d")
#imdtr2$treatment <- sub(imdtr2$treatment, pattern = "HT[0-9]-",
   replacement = "")
old <- unique(imdtr2$treatment)
new <- c("2d", "0.5d", "1d")
imdtr2$treatment[imdtr2$treatment %in% old] <- new[match(imdtr2$treatment,
    old)]
```

Now merge:

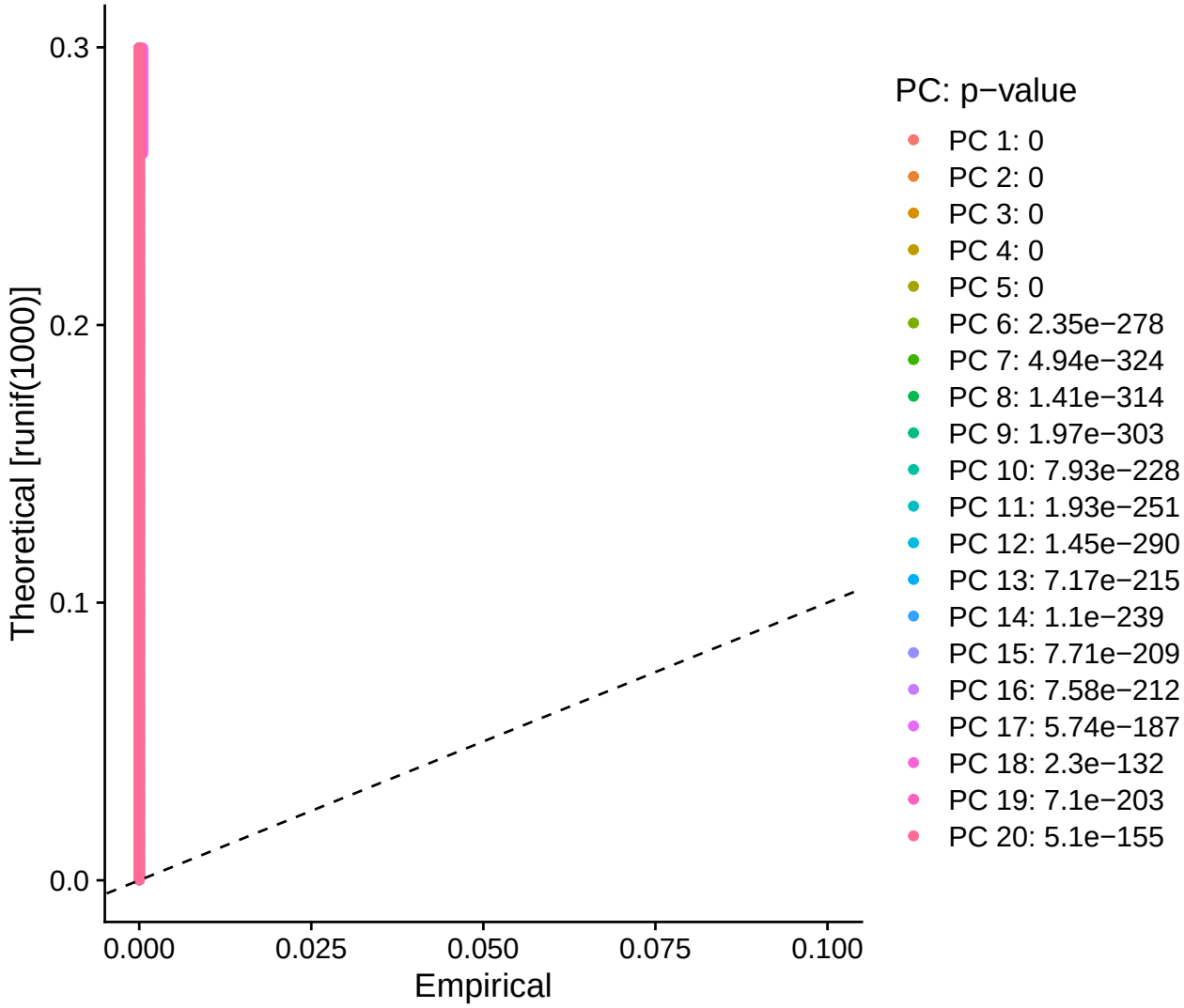
```
imdtr3 <- merge(imdtr1, imdtr2, project = "IM-DTR")
```

3.2 Normalize and process:

```
imdtr3 <- NormalizeData(imdtr3)
imdtr3 <- FindVariableFeatures(imdtr3, selection.method = "vst", nfeatures
    = 2000)
imdtr3 <- ScaleData(imdtr3, features = rownames(imdtr3))
```

```
imdtr3 <- RunPCA(imdtr3, features = VariableFeatures(imdtr3))
```

```
imdtr3 <- JackStraw(imdtr3, num.replicate = 100)
imdtr3 <- ScoreJackStraw(imdtr3, dims = 1:20)
JackStrawPlot(imdtr3, dims = 1:20)
```



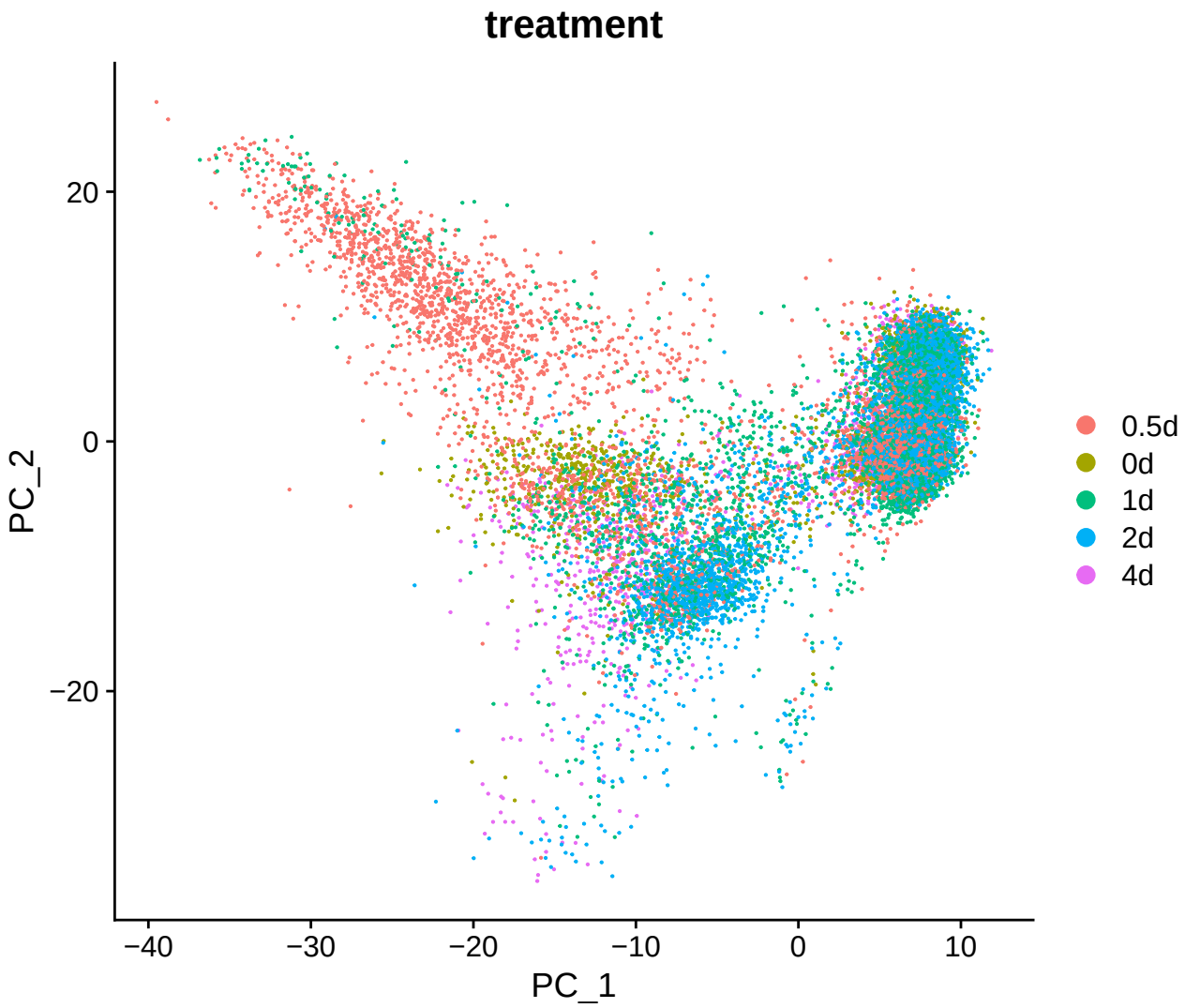
Samples from Exp1 and Exp2 are so similar.

Remove old calculated neighbor (snn):

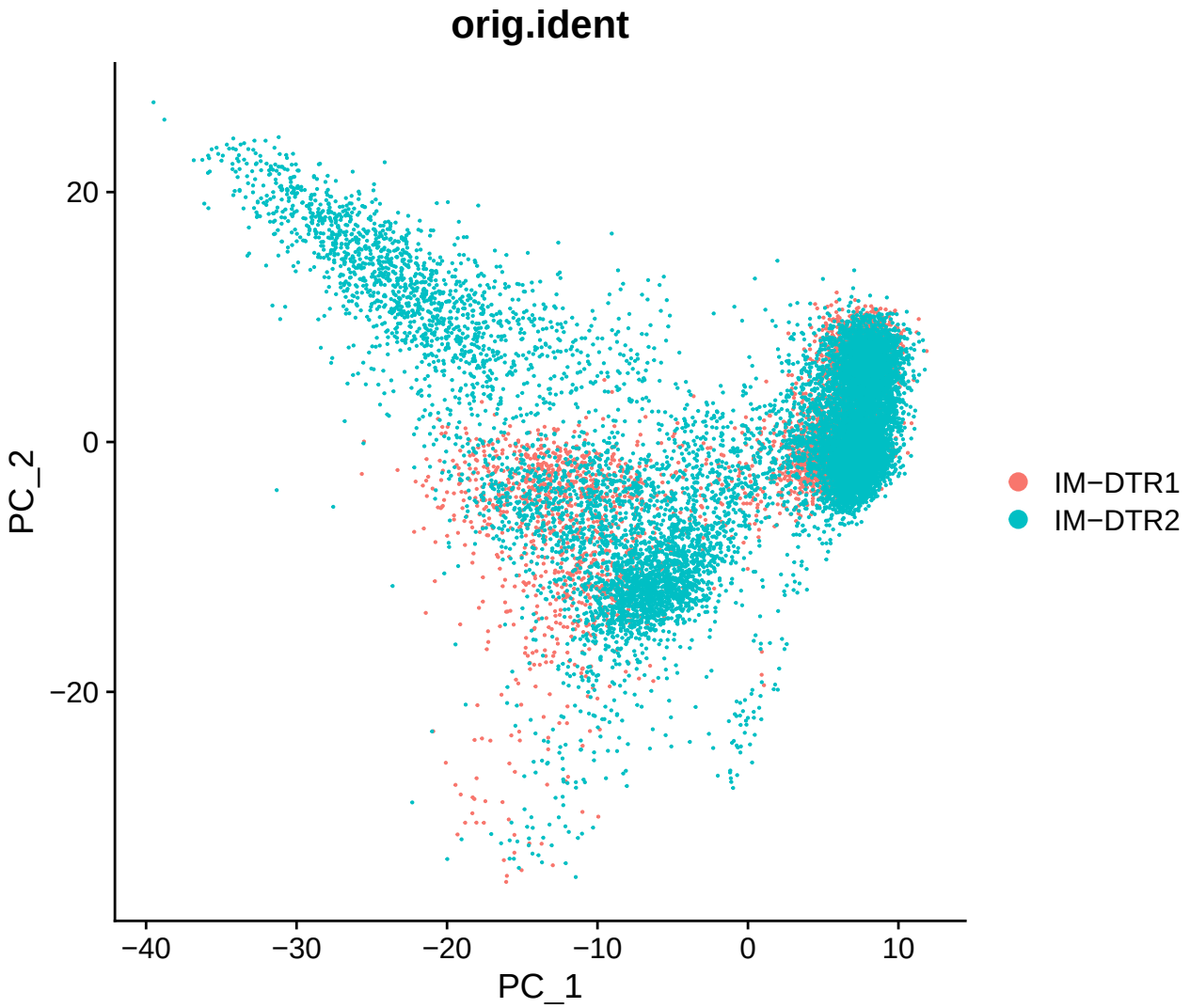
```
meta.names <- names(imdtr3@meta.data)
old.snn <- meta.names[startsWith(meta.names, "RNA_snn") | startsWith(meta.
    names, "res") | startsWith(meta.names, "Clusternames") | startsWith(
    meta.names, "ClusterNames")]
for (i in old.snn) {
  imdtr3[[i]] <- NULL
}
```

```
imdtr3 <- RunPCA(imdtr3, features = VariableFeatures(imdtr3))
```

```
PCAPlot(imdtr3, group.by = "treatment")
```




```
PCAPlot(imdtr3, group.by = "orig.ident")
```



Samples from Exp1 and Exp2 are so similar.

3.3 Non-linear reduction:

```
results <- RunTSNE(imdtr3, dims = 1:6)
```

Calculate UMAP in 3D:

```
imdtr3.for3d <- RunUMAP(imdtr3, dims = 1:6, n.components = 3L)
```

3.4 Presentation in 3D

```
library(rgl)
library(RColorBrewer)

umap_1 <- imdtr3.for3d@reductions$umap@cell.embeddings[,1]
umap_2 <- imdtr3.for3d@reductions$umap@cell.embeddings[,2]
umap_3 <- imdtr3.for3d@reductions$umap@cell.embeddings[,3]

```

```
#plot3d(x = umap_1, y = umap_2, z = umap_3, col = factor(imdtr3.for3d$cell
    .type2, labels = brewer.pal(length(unique(imdtr3.for3d$cell.type2)),
    name = "Paired")), size = 3)
#rglwidget()
```

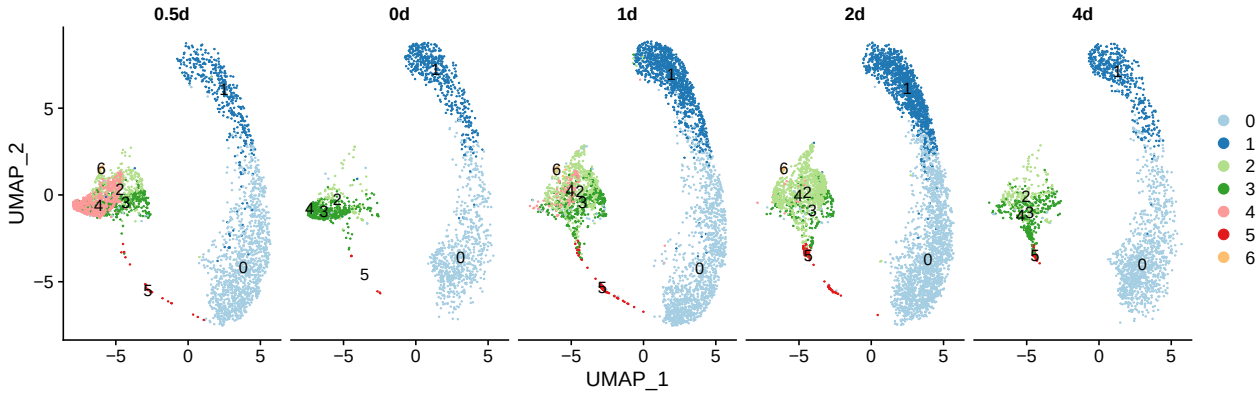
3d plot shows only one bridge exists.

3.5 Cluster cells

```
imdtr3.for3d <- FindNeighbors(imdtr3.for3d, dims = 1:10)
imdtr3.for3d <- FindClusters(imdtr3.for3d, resolution = 0.23)
```

```
## Modularity Optimizer version 1.3.0 by Ludo Waltman and Nees Jan van Eck
##
## Number of nodes: 15941
## Number of edges: 508837
##
## Running Louvain algorithm...
## Maximum modularity in 10 random starts: 0.9276
## Number of communities: 7
## Elapsed time: 1 seconds
```

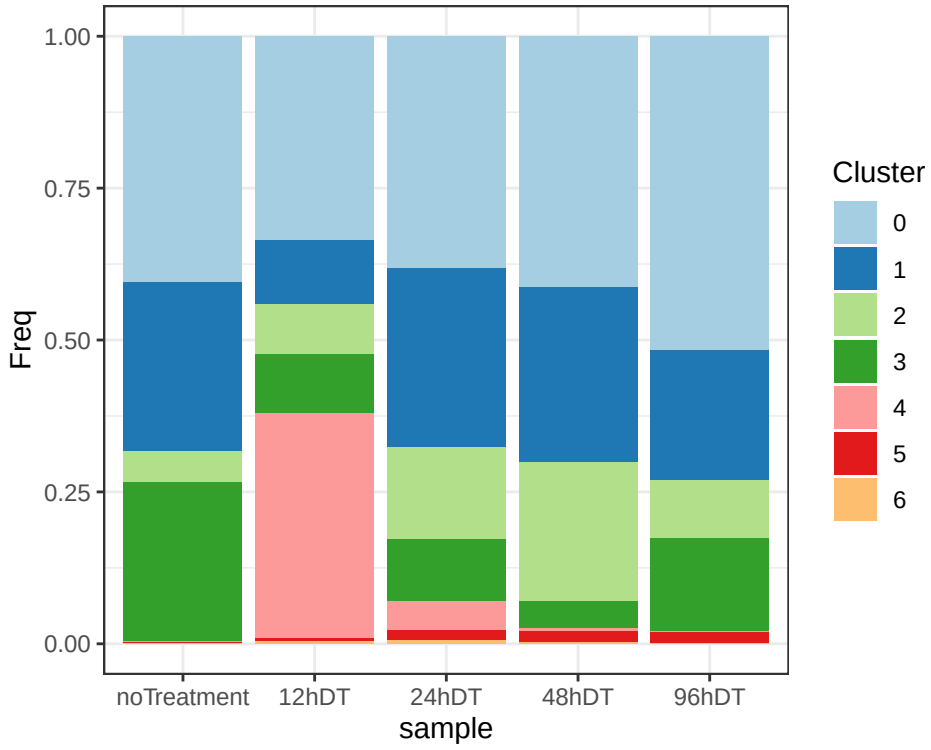
```
Idents(imdtr3.for3d) <- "RNA_snn_res.0.23"
pal <- brewer.pal(length(levels(imdtr3.for3d)), name = "Paired")
DimPlot(imdtr3.for3d, label = TRUE, split.by = "treatment", cols = pal,
    reduction = "umap")
```



The cluster 6 could be DCs

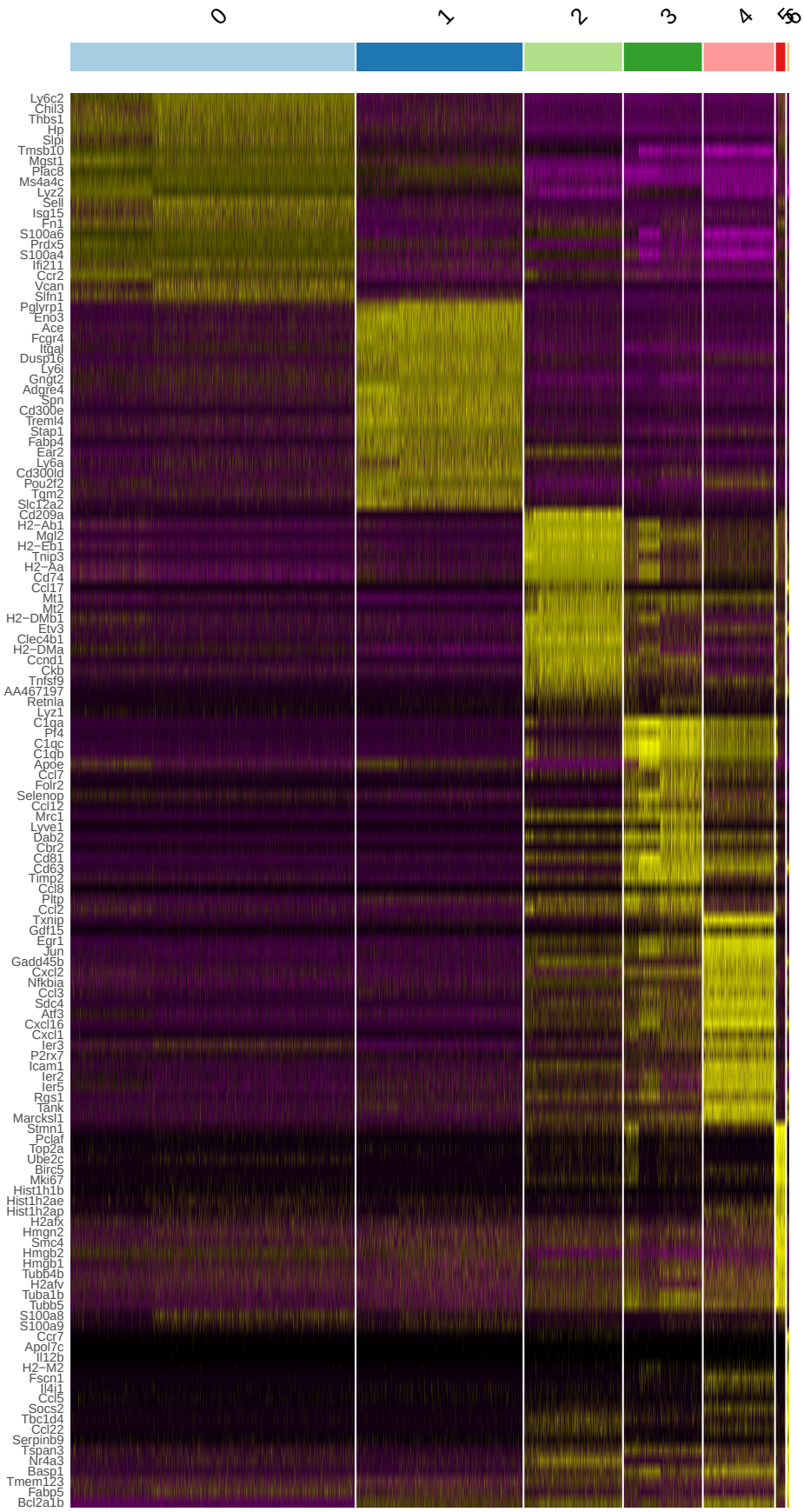
```
source("../R/SeuratFreqTable.R")
freq.celltype.list <- list(
  `noTreatment` = Seurat2CellFreqTable(subset(imdtr3.for3d, subset =
     treatment == "0d"), slotName = "RNA_snn_res.0.23"),
  `12hDT` = Seurat2CellFreqTable(subset(imdtr3.for3d, subset = treatment
     == "0.5d"), slotName = "RNA_snn_res.0.23"),
  `24hDT` = Seurat2CellFreqTable(subset(imdtr3.for3d, subset = treatment
     == "1d"), slotName = "RNA_snn_res.0.23"),
  `48hDT` = Seurat2CellFreqTable(subset(imdtr3.for3d, subset = treatment
     == "2d"), slotName = "RNA_snn_res.0.23"),
  `96hDT` = Seurat2CellFreqTable(subset(imdtr3.for3d, subset = treatment
     == "4d"), slotName = "RNA_snn_res.0.23")
```

```
)

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



4 Population characterization

```
library(dplyr)
all_cluster.markers <- FindAllMarkers(imdtr3.for3d, verbose = FALSE)
top20 <- all_cluster.markers %>% group_by(cluster) %>% top_n(n = 20, wt =
    avg_log2FC)
DoHeatmap(imdtr3.for3d, features = top20$gene, group.colors = pal) +
    NoLegend()
```



Cluster 0: ClaMo Cluster 1:

PatMo Cluster 2: MHCII_IM Cluster 3: CD206_IM Cluster 4: Chemoattractant Cluster 5: Transit Cluster 6: ?

4.1 Characterization of Cluster 6

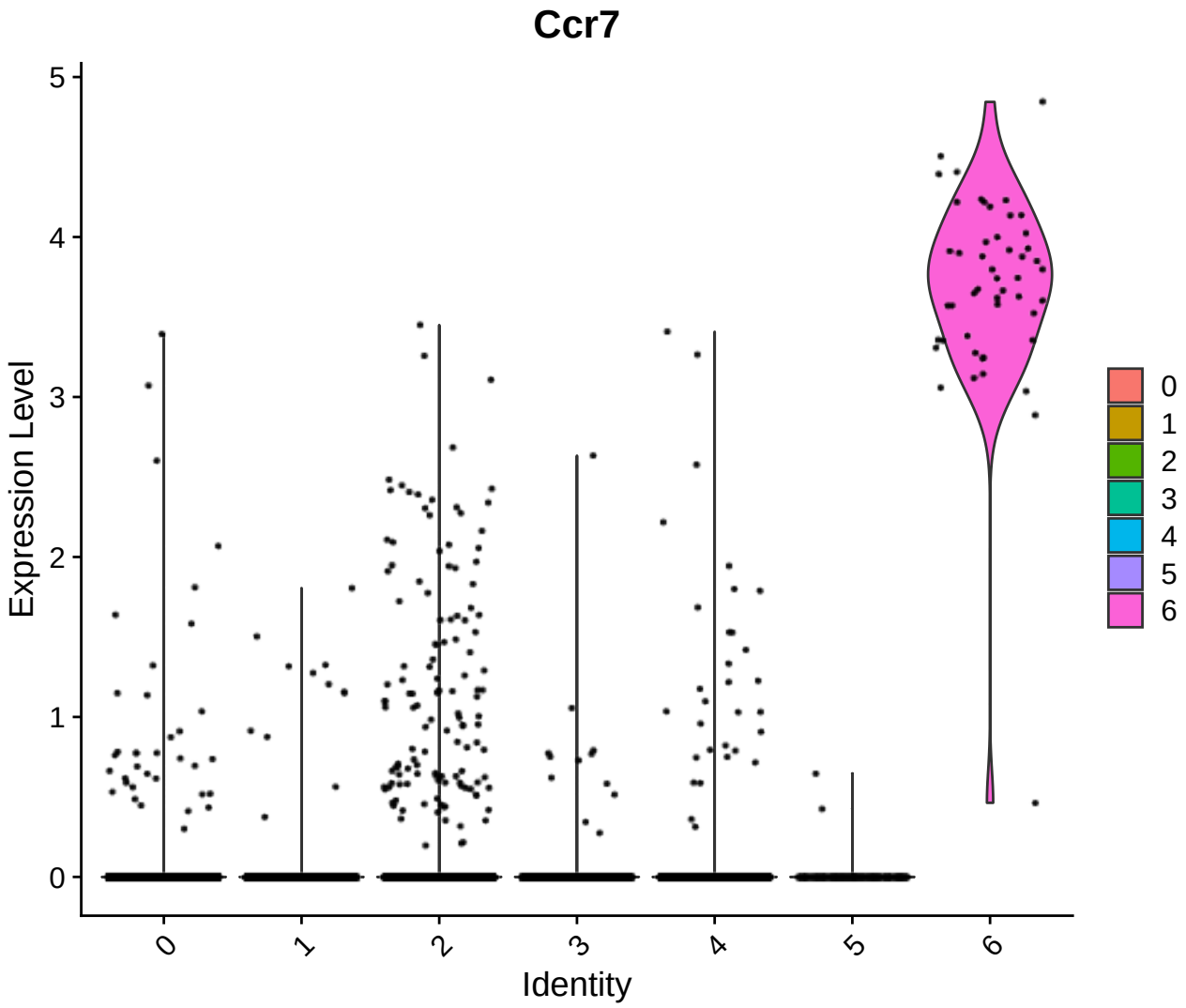
GO enrichment for cluster 6:

```
library(topGO)
library(org.Mm.eg.db)
allSymbols <- unique(AnnotationDbi::select(org.Mm.eg.db, keys(org.Mm.eg.db),columns = "SYMBOL")$SYMBOL)
extended_signatures_cluster6 <- filter(all_cluster.markers, avg_log2FC > 0.5 & -log10(p_val) > 2 & cluster == "6")
myInterestingGenes <-
  as.character(extended_signatures_cluster6$gene)

geneList <- factor(as.integer(allSymbols %in% myInterestingGenes))
names(geneList) <- allSymbols


sampleGOdata <- new("topGOdata",
                    description = "Simple session", ontology = "BP",
                    allGenes = geneList,
                    annot = annFUN.org, mapping = "org.Mm.eg.db", ID = "SYMBOL")
resultFisher <- runTest(sampleGOdata, algorithm = "classic", statistic = "fisher")
GenTable(sampleGOdata, classicFisher = resultFisher, topNodes = 20)
```

```
## # A tibble: 20 x 6
##    GO.ID      Term                  Annotated Significant Expected      classicFisher
##    <chr>      <chr>                     <int>       <int>    <dbl> <chr>
##  1 GO:0048523 negative regulation ~      4950         173     77.2 2.4e-29
##  2 GO:0048519 negative regulation ~      5500         181     85.8 8.5e-28
##  3 GO:0009987 cellular process          17114         343    267.  2.6e-25
##  4 GO:0048518 positive regulation ~      6358         189     99.2 9.8e-24
##  5 GO:0065009 regulation of molecu~      2574         109     40.2 3.6e-23
##  6 GO:0050789 regulation of biolog~     12212         280    191.  1.6e-22
##  7 GO:0050790 regulation of cataly~      1876          89     29.3 7.2e-22
##  8 GO:0048522 positive regulation ~      5834         175     91.0 9.1e-22
##  9 GO:0050794 regulation of cellul~     11652         270    182.  1.6e-21
## 10 GO:0002376 immune system process      2832         112     44.2 1.8e-21
```

```
## 11 GO:0065007 biological regulation     12806          286     200.  1.9e-21
## 12 GO:0040011 locomotion                 1863           87      29.1 6.6e-21
## 13 GO:0016477 cell migration             1501           77      23.4 8.3e-21
## 14 GO:0048870 cell motility              1663           81      26.0 1.6e-20
## 15 GO:0051674 localization of cell       1663           81      26.0 1.6e-20
## 16 GO:0048856 anatomical structure~      5977          174      93.3 3.9e-20
## 17 GO:0051336 regulation of hydrol~      1002           61      15.6 4.7e-20
## 18 GO:0032502 developmental process      6449          182     101.  7.7e-20
## 19 GO:0006928 movement of cell or ~      2101           91      32.8 1.0e-19
## 20 GO:0048583 regulation of respon~      3972          133      62.0 1.9e-19
```

```
VlnPlot(imdtr3.for3d, features = "Ccr7")
```



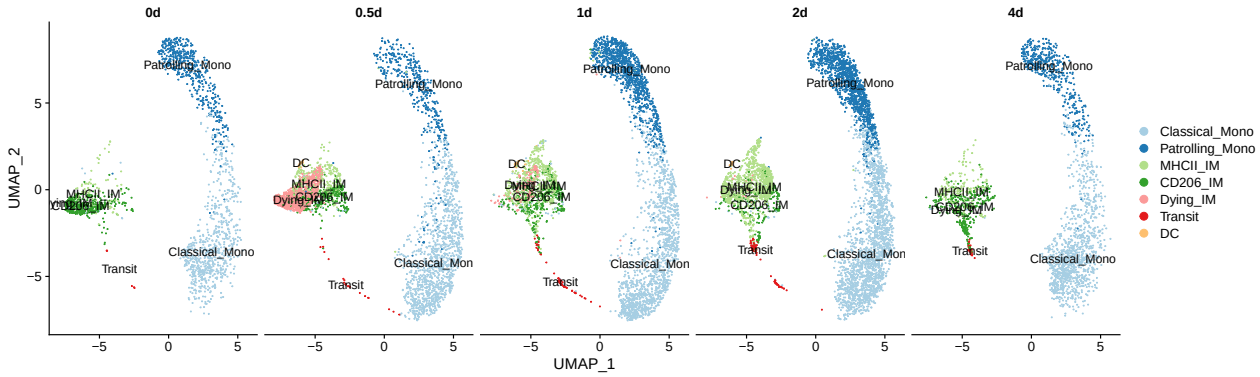
Cluster 0: ClaMo Cluster 1: PatMo Cluster 2: MHCII_IM Cluster 3: CD206_IM Cluster 4: Dying_IM Cluster 5: Transit Cluster 6: DC

change the idents

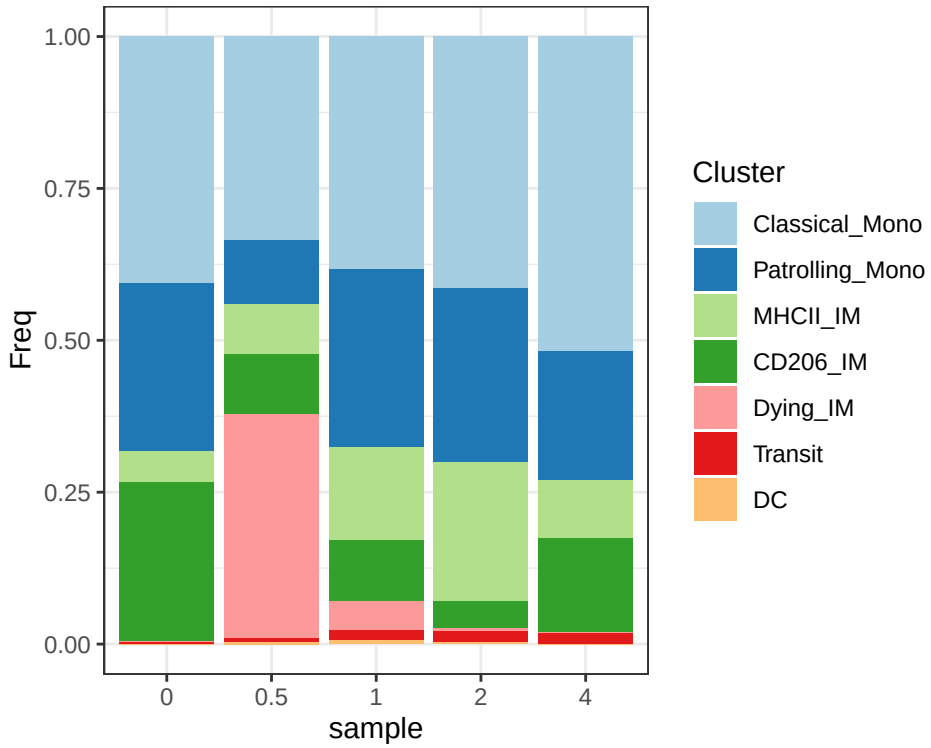
```
levels(imdtr3.for3d)
```

```
## [1] "0" "1" "2" "3" "4" "5" "6"
```

```
imdtr3.for3d$cell.type2 <- factor(Idents(imdtr3.for3d), labels = c("
   Classical_Mono", "Patrolling_Mono", "MHCII_IM", "CD206_IM", "Dying_IM",
    "Transit", "DC"))


Idents(imdtr3.for3d) <- "cell.type2"
imdtr3.for3d$treatment <- factor(imdtr3.for3d$treatment, levels = c("0d",
   "0.5d", "1d", "2d", "4d"))
DimPlot(imdtr3.for3d, label = TRUE, split.by = "treatment", cols = pal)
```




```
freq.celltype.list <- list(
  `0` = Seurat2CellFreqTable(subset(imdtr3.for3d, subset = treatment == "0
     d"), slotName = "cell.type2"),
  `0.5` = Seurat2CellFreqTable(subset(imdtr3.for3d, subset = treatment ==
     "0.5d"), slotName = "cell.type2"),
  `1` = Seurat2CellFreqTable(subset(imdtr3.for3d, subset = treatment == "1
     d"), slotName = "cell.type2"),
  `2` = Seurat2CellFreqTable(subset(imdtr3.for3d, subset = treatment == "2
     d"), slotName = "cell.type2"),
  `4` = Seurat2CellFreqTable(subset(imdtr3.for3d, subset = treatment == "4
     d"), slotName = "cell.type2")
)

barChart(freq.celltype.list) + labs(fill = "Cluster") + scale_fill_manual(
    values = pal)
```

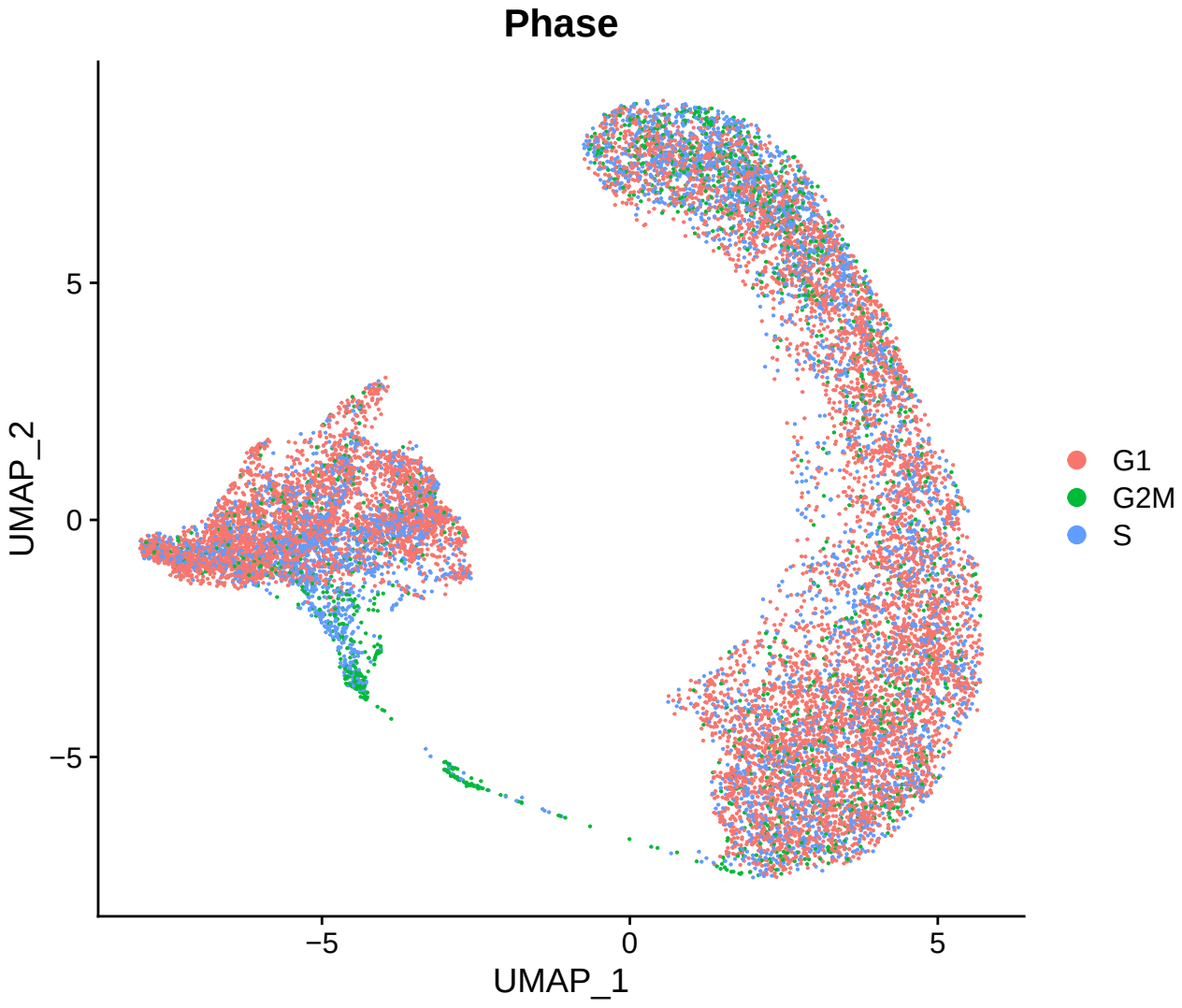



```
ggsave(filename = "BarPlot_distribution_celltypes_in_samples.pdf",
```

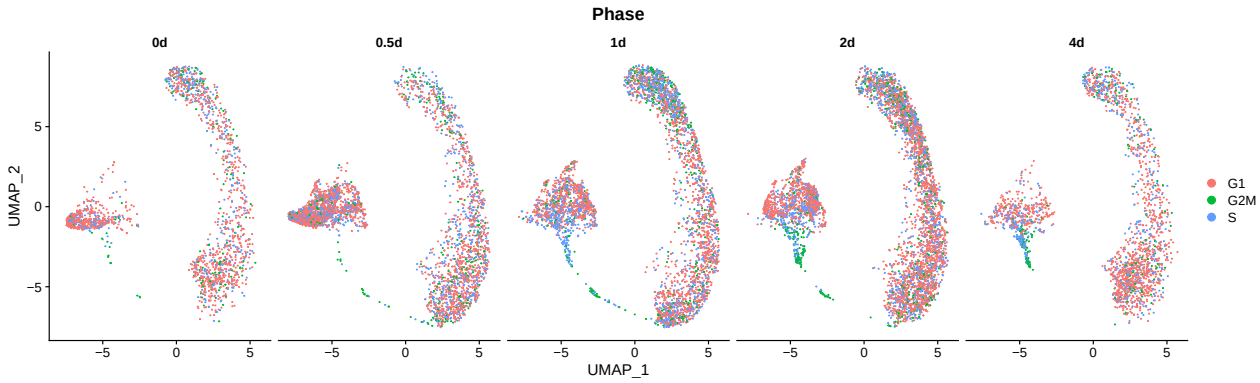
```
        path = "../Figures",
        height = 4, width = 5,
        device = "pdf")
```

4.2 Cell-cycle analysis

```
library(cowplot)
data("geneinfo_human", package = "nichenetr")
s.genes <- nichenetr::convert_human_to_mouse_symbols(cc.genes.updated.2019
    $s.genes)
g2m.genes <- nichenetr::convert_human_to_mouse_symbols(cc.genes.updated
    .2019$g2m.genes)
imdtr3.for3d <- CellCycleScoring(imdtr3.for3d, s.features = s.genes, g2m.
    features = g2m.genes, set.ident = FALSE)
DimPlot(imdtr3.for3d, group.by = "Phase")
```

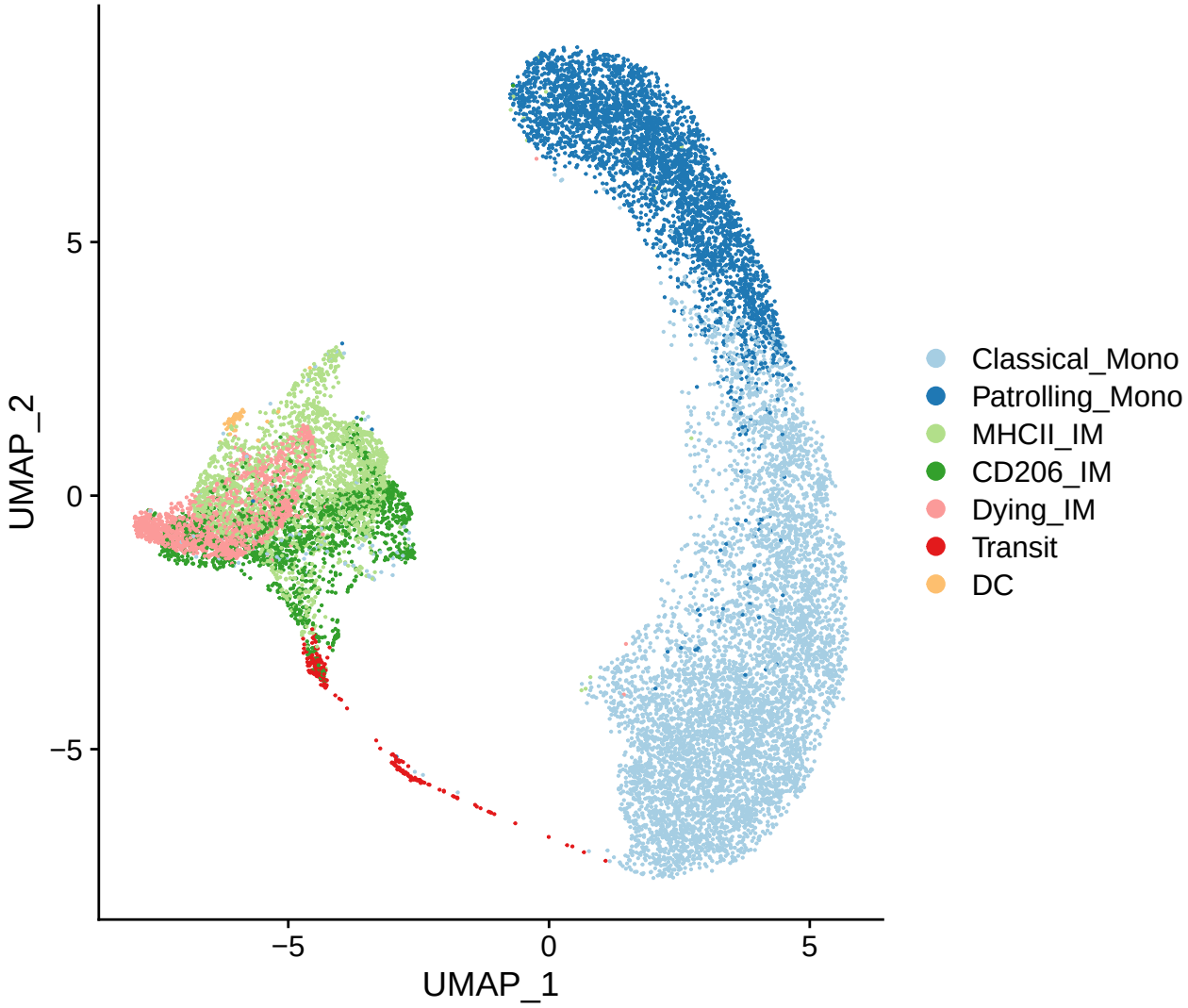



```
DimPlot(imdtr3.for3d, group.by = "Phase", reduction = "umap", split.by = "
    treatment")
```



2d plot:

```
DimPlot(imdtr3.for3d, cols = pal)
```
1

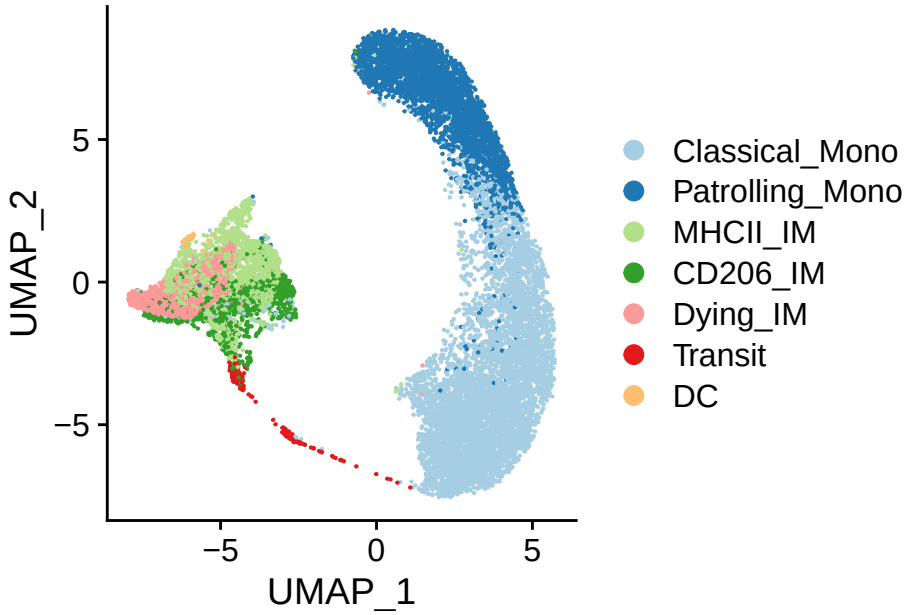



```
saveRDS(imdtr3.for3d, file = "./immune_imdtr3.seuratObject.Rds")
```
1

5 Generate plots

2D plot (component 1 and 2):

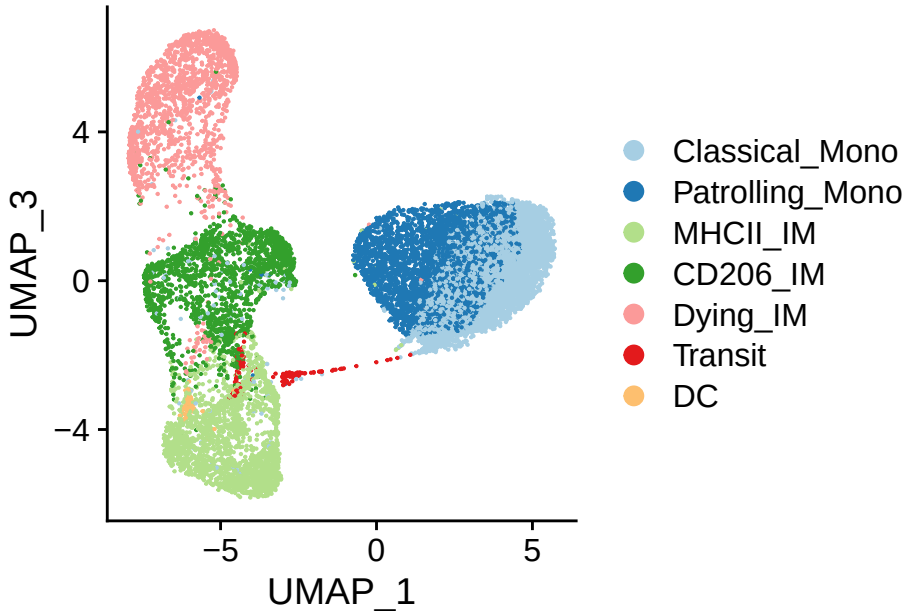
```
DimPlot(imdtr3.for3d, cols = pal, dims = c(1,2))
```




```
ggsave(filename = "UMAPplot_All_cells_PC1_2.pdf",
       path = "../Figures",
       height = 3.5, width = 5,
       device = "pdf")
```

2D plot (component 1 and 3):

```
DimPlot(imdtr3.for3d, cols = pal, dims = c(1,3))
```



```
ggsave(filename = "UMAPplot_All_cells_PC1_3.pdf",
       path = "../Figures",
       height = 3.5, width = 5,
       device = "pdf")
```

3D plot:

```
umap_1 <- imdtr3.for3d@reductions$umap@cell.embeddings[,1]
umap_2 <- imdtr3.for3d@reductions$umap@cell.embeddings[,2]
umap_3 <- imdtr3.for3d@reductions$umap@cell.embeddings[,3]
```

```
# open3d()
points3d(x = umap_1, y = umap_2, z = umap_3,
       col = factor(Idents(imdtr3.for3d), labels = brewer.pal(length(
          unique(imdtr3.for3d$cell.type2)), name = "Paired")),
       size = 7.5, alpha = 0.8, point_antialias = TRUE,
  line_antialias = TRUE)
#decorate3d(  box = FALSE,axes = FALSE, xlab = NULL, ylab = NULL, zlab =
   NULL)
#axes3d(edges = c("x--", "y--", "z--"), alpha = 0.5)
# bbox3d(box=TRUE)
#title3d(xlab = 'UMAP1', ylab = 'UMAP2', zlab = 'UMAP3', alpha = 0.7)
rgl.bringtotop()
rgl.viewpoint(30, -20, zoom = 0.7)
# rgl.postscript( filename = "../Docs/Figures/test.eps", fmt = "eps",
   drawText = TRUE )
# rglwidget()
#close3d()
```

```
# open3d()
points3d(x = umap_1, y = umap_2, z = umap_3,
       col = factor(Idents(imdtr3.for3d), labels = brewer.pal(length(
          unique(imdtr3.for3d$cell.type2)), name = "Paired")),
       size = 7.5, alpha = 0.8, point_antialias = TRUE,
  line_antialias = TRUE)
#decorate3d(  box = FALSE,axes = FALSE, xlab = NULL, ylab = NULL, zlab =
   NULL)
axes3d(edges = c("x--", "y--", "z--"), alpha = 1)
# bbox3d(box=TRUE)
#title3d(xlab = 'UMAP1', ylab = 'UMAP2', zlab = 'UMAP3', alpha = 0.7)
rgl.bringtotop()
rgl.viewpoint(30, -20, zoom = 0.7)
# rgl.postscript( filename = "../Docs/Figures/test.eps", fmt = "eps",
   drawText = TRUE )
# rglwidget()
#close3d()
```

```
# open3d()
spheres3d(x = umap_1, y = umap_2, z = umap_3,
       col = factor(Idents(imdtr3.for3d), labels = brewer.pal(length(
          unique(imdtr3.for3d$cell.type2)), name = "Paired")),
       radius = 0.1, alpha = 0.5,
       # ambient      = "gray",
```

```
       #emission     = "gray",
       specular     = "gray",
       shininess = 10)
#decorate3d(  box = FALSE,axes = FALSE, xlab = NULL, ylab = NULL, zlab =
   NULL)
#axes3d(edges = c("x--", "y--", "z--"), alpha = 0.5)
# bbox3d(box=TRUE)
#title3d(xlab = 'UMAP1', ylab = 'UMAP2', zlab = 'UMAP3', alpha = 0.7)
rgl.bringtotop()
rgl.viewpoint(30, -20, zoom = 0.7)
# rgl.postscript( filename = "../Docs/Figures/test.eps", fmt = "eps",
   drawText = TRUE )
# rglwidget()
#close3d()
```

Make snapshots for every sample:

```
plot3d.byGroup <- function(seuratObj, treat, savePNG=FALSE, ...) {
obj <- subset(seuratObj, subset = treatment == treat)
col.pal <- factor(Idents(obj), labels = brewer.pal(length(unique(obj$cell.
   type2)), name = "Paired"))


umap_1 <- obj@reductions$umap@cell.embeddings[,1]
umap_2 <- obj@reductions$umap@cell.embeddings[,2]
umap_3 <- obj@reductions$umap@cell.embeddings[,3]

umap1.min <- min(seuratObj@reductions$umap@cell.embeddings[,1])
umap1.max <- max(seuratObj@reductions$umap@cell.embeddings[,1])

umap2.min <- min(seuratObj@reductions$umap@cell.embeddings[,2])
umap2.max <- max(seuratObj@reductions$umap@cell.embeddings[,2])

umap3.min <- min(seuratObj@reductions$umap@cell.embeddings[,3])
umap3.max <- max(seuratObj@reductions$umap@cell.embeddings[,3])

# scatterplot3d(x = umap_1, y = umap_2, z = umap_3, # old one
#               color = col.pal, pch = 20,
#               grid=FALSE, box=TRUE, col.axis="black",
#               main = treat,
#               ...)
# open3d()
points3d(x = umap_1, y = umap_2, z = umap_3,
       col = col.pal,
       size = 7.5, alpha = 0.8,
       point_antialias = TRUE,
  line_antialias = TRUE)

# decorate3d(xlim = c(umap1.min, umap1.max),
#            ylim = c(umap2.min, umap2.max),
#            zlim = c(umap3.min, umap3.max),
#            xlab = NULL, ylab = NULL, zlab = NULL,
#            box = FALSE,axes = FALSE)
# bbox3d(box=TRUE)
```

```
#title3d(xlab = 'UMAP1', ylab = 'UMAP2', zlab = 'UMAP3', alpha = 0.7)
rgl.bringtotop()
#rgl.viewpoint(3, 180)
# rgl.postscript( filename = "../Docs/Figures/test.eps", fmt = "eps",
    drawText = TRUE )
rgl.viewpoint(30, -20, zoom = 0.7)

if (savePNG) {
  filename <- file.path("../Docs/Figures", paste(treat, ".png", sep = ""))
  png(filename)
  rglwidget()
  dev.off()
} else (return(rglwidget()))
#close3d()
}
```

```
treat.list <- c("0.5d","1d","2d","4d","0d")
# everytime, save PNG manually with width = 700, height = 630.
#  plot3d.byGroup(imdtr3.for3d, treat.list[1])
```

```
#plot3d.byGroup(imdtr3.for3d, treat.list[2])
```

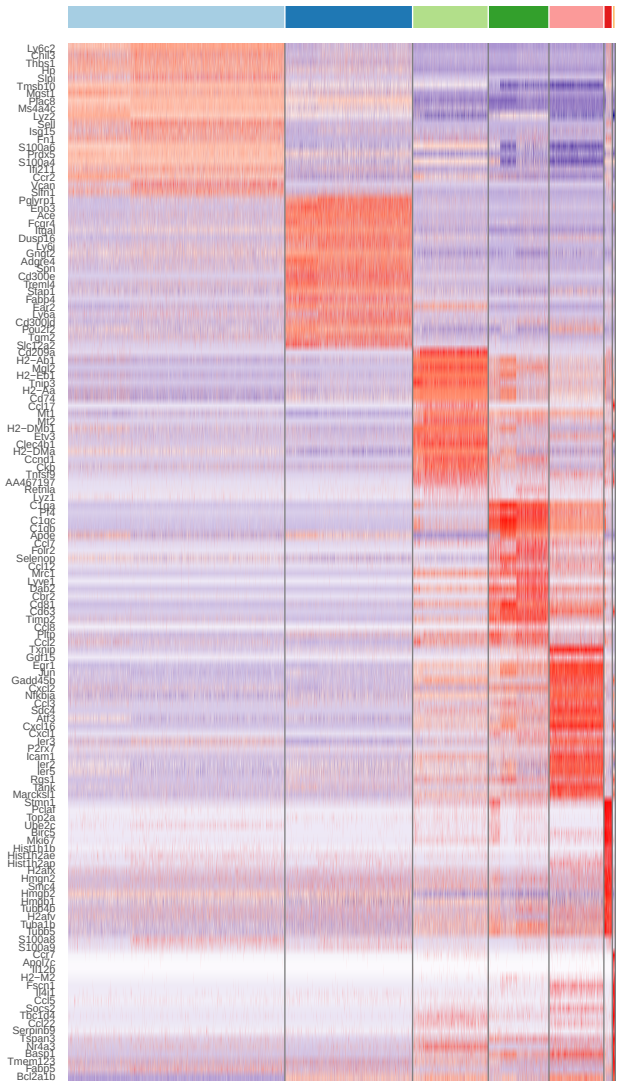
```
#plot3d.byGroup(imdtr3.for3d, treat.list[3])
```

```
#plot3d.byGroup(imdtr3.for3d, treat.list[4])
```

```
#plot3d.byGroup(imdtr3.for3d, treat.list[5])
```

Heatmap:

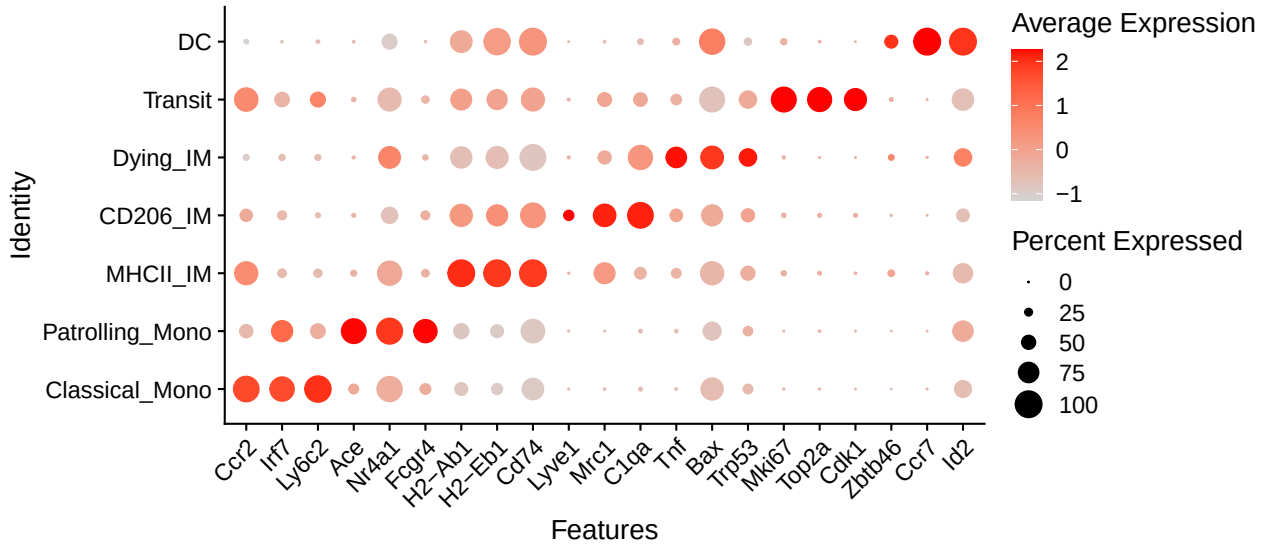
```
DoHeatmap(imdtr3.for3d, features = top20$gene, group.colors = pal, size =
    0) + scale_fill_gradientn(colors = c("dark blue", "white", "red")) +
    theme(axis.text.y = element_text(size = 4))
```




```
ggsave(filename = "HeatMap_top20_in_cell_types_blue_red.pdf",
       path = "../Figures",
       height = 6, width = 5,
       device = "pdf")
```

Dot plot show cell type markers:

```
DotPlot(imdtr3.for3d,
        cols = c("light grey", "red"),
        features = c("Ccr2", "Irf7", "Ly6c2",
                     "Ace", "Nr4a1", "Fcgr4",
                     "H2-Ab1", "H2-Eb1", "Cd74",
                     "Lyve1", "Mrc1", "C1qa",
                     "Tnf", "Bax", "Trp53",
                     "Mki67", "Top2a", "Cdk1",
                     "Zbtb46", "Ccr7", "Id2"
                     )) +
   theme(axis.text.x = element_text(angle = 45, hjust = 1))
```




```
ggsave(filename = "DotPlot_Markers_in_cell_types.pdf",
       path = "../Figures",
       height = 4, width = 9,
       device = "pdf")
```

Cell-cycle score:

```
# color gradient
col_gradient <- colorRampPalette(c("lightgrey", "red"))(256)

# assign color in scaled cell-cycle score
val <- imdtr3.for3d$G2M.Score
col_ix <- round((val - min(val))/(max(val)-min(val)) * 255, digits = 0)
col_pal <- col_gradient[col_ix + 1 ] # because range will be 0:255
names(col_pal) <- names(col_ix)

# remake order to avoid overlay
umap_1.order_g2m <- umap_1[order(imdtr3.for3d$G2M.Score, decreasing = TRUE
   )]
umap_2.order_g2m <- umap_2[order(imdtr3.for3d$G2M.Score, decreasing = TRUE
   )]
umap_3.order_g2m <- umap_3[order(imdtr3.for3d$G2M.Score, decreasing = TRUE
   )]
col_pal <- col_pal[order(imdtr3.for3d$G2M.Score, decreasing = TRUE)]
```

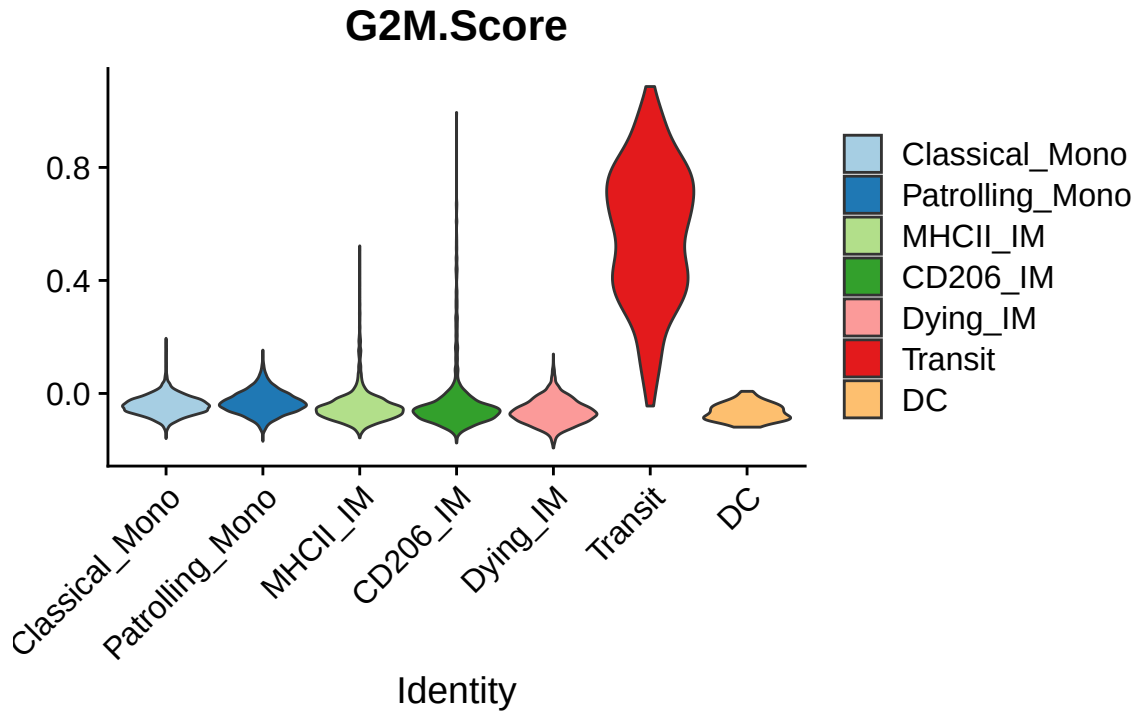
```
#open3d()
points3d(x = umap_1.order_g2m, y = umap_2.order_g2m, z = umap_3.order_g2m,
       col = col_pal,
       size = 7.5, alpha = 1,
       point_antialias = TRUE,
  line_antialias = TRUE)
#decorate3d(  box = FALSE,axes = FALSE, xlab = NULL, ylab = NULL, zlab =
   NULL)
#axes3d(edges = c("x--", "y--", "z--"), alpha = 0.5)
# bbox3d(box=TRUE)
#title3d(xlab = 'UMAP1', ylab = 'UMAP2', zlab = 'UMAP3', alpha = 0.7)
```

```
rgl.bringtotop()
rgl.viewpoint(30, -20, zoom = 0.7)
# rgl.postscript( filename = "../Docs/Figures/test.eps", fmt = "eps",
    drawText = TRUE )
#rglwidget()
#close3d()
```

```
#open3d()
points3d(x = umap_1, y = umap_2, z = umap_3,
       col = factor(Idents(imdtr3.for3d), labels = brewer.pal(length(
          unique(imdtr3.for3d$cell.type2)), name = "Paired")),
       size = 7.5, alpha = 1,
       point_antialias = TRUE,
  line_antialias = TRUE)
#decorate3d(  box = FALSE,axes = FALSE, xlab = NULL, ylab = NULL, zlab =
    NULL)
#axes3d(edges = c("x--", "y--", "z--"), alpha = 0.5)
# bbox3d(box=TRUE)
#title3d(xlab = 'UMAP1', ylab = 'UMAP2', zlab = 'UMAP3', alpha = 0.7)
rgl.bringtotop()
rgl.viewpoint(30, -20, zoom = 0.7)
# rgl.postscript( filename = "../Docs/Figures/test.eps", fmt = "eps",
    drawText = TRUE )
#rglwidget()
#close3d()
```

Score:

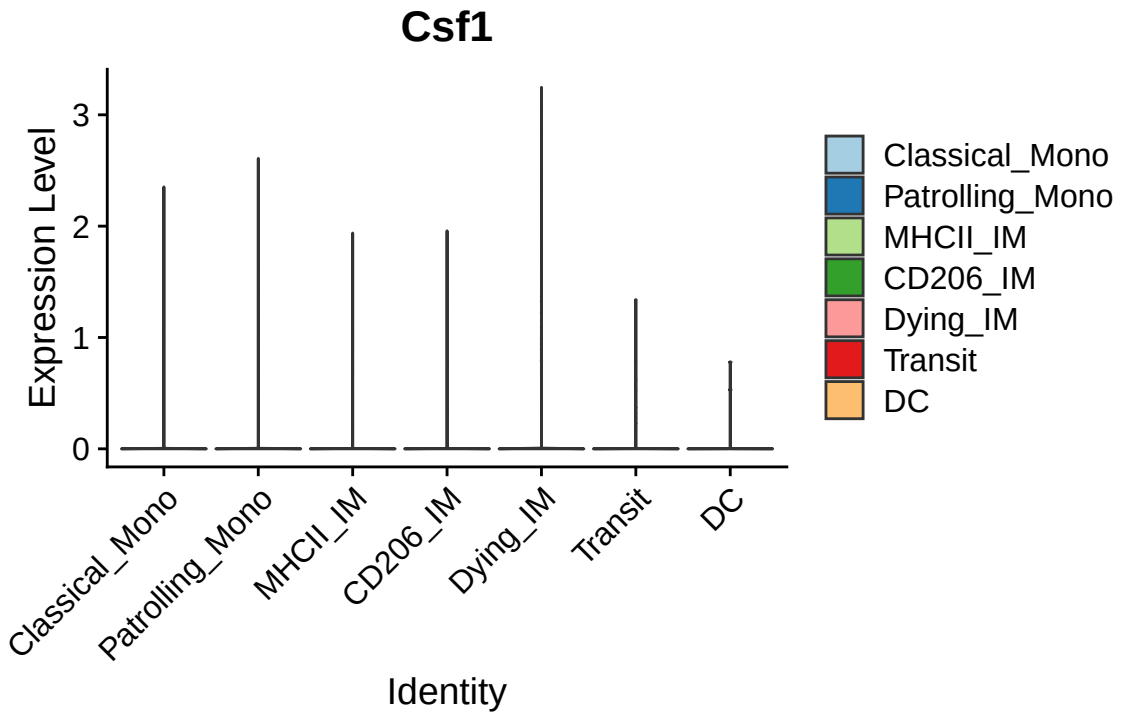
```
VlnPlot(imdtr3.for3d, features = "G2M.Score", cols = pal, pt.size = 0)
```



```
ggsave(filename = "VlnPlot_G2MScore_in_all_samples.pdf",
       path = "../Figures",
       height = 4, width = 6,
       device = "pdf")
```

Csf1 expression:

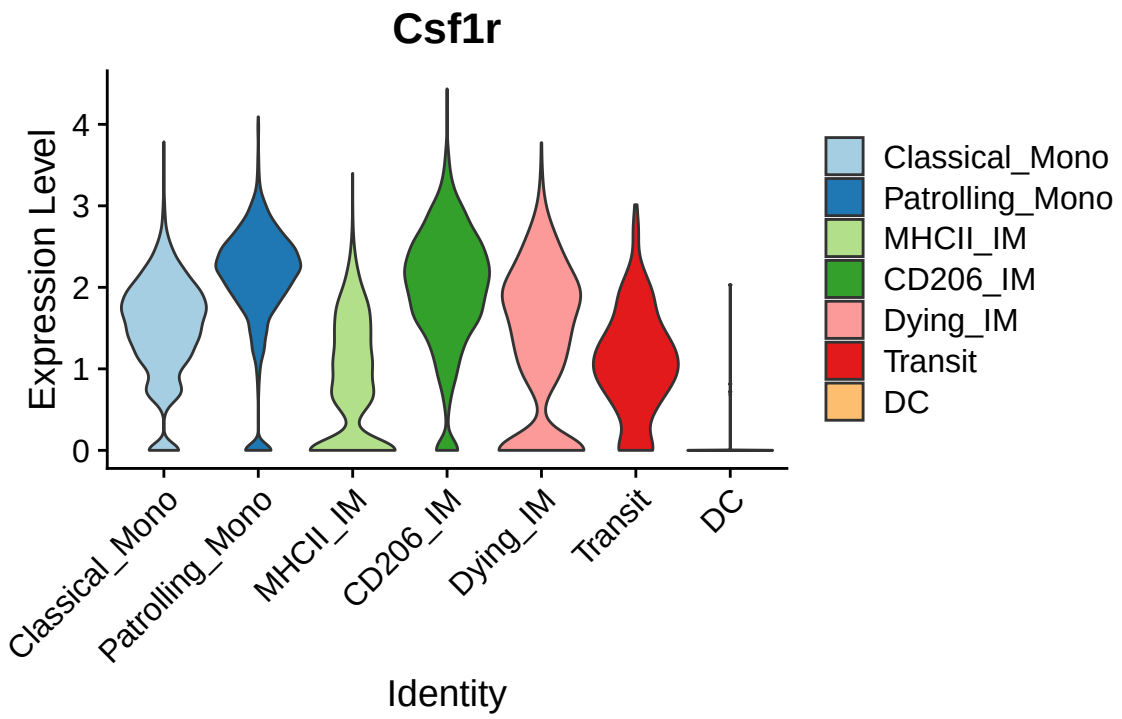
```
VlnPlot(imdtr3.for3d, features = "Csf1", cols = pal, pt.size = 0)
```




```
ggsave(filename = "VlnPlot_Csf1_in_all_samples.pdf",
       path = "../Figures",
       height = 4, width = 6,
       device = "pdf")
```

Csf1r

```
VlnPlot(imdtr3.for3d, features = "Csf1r", cols = pal, pt.size = 0)
```




```
ggsave(filename = "VlnPlot_Csf1r_in_all_samples.pdf",
       path = "../Figures",
       height = 4, width = 6,
       device = "pdf")
```

6 Session information

```
sessionInfo()
```

```
## R version 4.0.3 (2020-10-10)
## Platform: x86_64-pc-linux-gnu (64-bit)
## Running under: Ubuntu 20.04.3 LTS
##
## Matrix products: default
## BLAS:   /usr/lib/x86_64-linux-gnu/openblas-pthread/libblas.so.3
## LAPACK: /usr/lib/x86_64-linux-gnu/openblas-pthread/liblapack.so.3
##
## locale:
##  [1] LC_CTYPE=en_US.UTF-8       LC_NUMERIC=C
##  [3] LC_TIME=en_GB.UTF-8        LC_COLLATE=en_US.UTF-8
##  [5] LC_MONETARY=en_GB.UTF-8    LC_MESSAGES=en_US.UTF-8
##  [7] LC_PAPER=en_GB.UTF-8       LC_NAME=C
##  [9] LC_ADDRESS=C               LC_TELEPHONE=C
## [11] LC_MEASUREMENT=en_GB.UTF-8 LC_IDENTIFICATION=C
##
## attached base packages:
## [1] stats4    parallel  stats     graphics  grDevices utils
    datasets
## [8] methods   base
##
```

```
## other attached packages:
##  [1] cowplot_1.1.1         org.Mm.eg.db_3.12.0   topGO_2.42.0
##  [4] SparseM_1.81          GO.db_3.12.1          AnnotationDbi_1.52.0
##  [7] IRanges_2.24.1        S4Vectors_0.28.1      Biobase_2.50.0
## [10] graph_1.68.0          BiocGenerics_0.36.1   dplyr_1.0.8
## [13] RColorBrewer_1.1-2    rgl_0.108.3           ggplot2_3.3.5
## [16] SeuratObject_4.0.4    Seurat_4.1.0
##
## loaded via a namespace (and not attached):
##   [1] utf8_1.2.2             reticulate_1.24        tidyselect_1.1.1
##   [4] RSQLite_2.2.10         htmlwidgets_1.5.4      grid_4.0.3
##   [7] Rtsne_0.15             pROC_1.18.0            munsell_0.5.0
##  [10] codetools_0.2-18       ica_1.0-2              future_1.24.0
##  [13] miniUI_0.1.1.1         withr_2.4.3            spatstat.random_2.1-0
##  [16] colorspace_2.0-3       highr_0.9              knitr_1.37
##  [19] rstudioapi_0.13        ROCR_1.0-11            tensor_1.5
##  [22] Rttf2pt1_1.3.10        listenv_0.8.0          labeling_0.4.2
##  [25] polyclip_1.10-0        bit64_4.0.5            farver_2.1.0
##  [28] parallelly_1.30.0      vctrs_0.3.8            generics_0.1.2
##  [31] ipred_0.9-12           xfun_0.29              randomForest_4.6-14
##  [34] R6_2.5.1               ggbeeswarm_0.6.0       bitops_1.0-7
##  [37] spatstat.utils_2.3-0   cachem_1.0.6           assertthat_0.2.1
##  [40] promises_1.2.0.1       scales_1.1.1           nnet_7.3-14
##  [43] beeswarm_0.4.0         gtable_0.3.0           Cairo_1.5-14
##  [46] globals_0.14.0         goftest_1.2-3          timeDate_3043.102
##  [49] rlang_1.0.1            splines_4.0.3          extrafontdb_1.0
##  [52] lazyeval_0.2.2         ModelMetrics_1.2.2.2   checkmate_2.0.0
##  [55] spatstat.geom_2.3-2    yaml_2.3.5             reshape2_1.4.4
##  [58] abind_1.4-5            backports_1.4.1        httpuv_1.6.5
##  [61] Hmisc_4.6-0            DiagrammeR_1.0.8       caret_6.0-90
##  [64] extrafont_0.17         tools_4.0.3            lava_1.6.10
##  [67] ellipsis_0.3.2         spatstat.core_2.4-0    proxy_0.4-26
##  [70] ggridges_0.5.3         Rcpp_1.0.8             plyr_1.8.6
##  [73] base64enc_0.1-3        visNetwork_2.1.0       purrr_0.3.4
##  [76] rpart_4.1-15           deldir_1.0-6           pbapply_1.5-0
##  [79] zoo_1.8-9              nichenetr_1.0.0        ggrepel_0.9.1
##  [82] cluster_2.1.0          magrittr_2.0.2         data.table_1.14.2
##  [85] RSpectra_0.16-0        scattermore_0.8        lmtest_0.9-39
##  [88] RANN_2.6.1             fitdistrplus_1.1-6     matrixStats_0.61.0
##  [91] hms_1.1.1              patchwork_1.1.1        mime_0.12
##  [94] evaluate_0.15          xtable_1.8-4           jpeg_0.1-9
##  [97] gridExtra_2.3          compiler_4.0.3         tibble_3.1.6
## [100] KernSmooth_2.23-20     crayon_1.5.0           htmltools_0.5.2
## [103] mgcv_1.8-33            later_1.3.0            tzdb_0.2.0
## [106] Formula_1.2-4          tidyr_1.2.0            lubridate_1.8.0
## [109] DBI_1.1.2              MASS_7.3-53            Matrix_1.4-0
## [112] readr_2.1.2            cli_3.2.0              gower_1.0.0
## [115] igraph_1.2.11          pkgconfig_2.0.3        foreign_0.8-82
## [118] plotly_4.10.0          spatstat.sparse_2.1-0  recipes_0.2.0
## [121] foreach_1.5.2          vipor_0.4.5            hardhat_0.2.0
## [124] prodlim_2019.11.13     stringr_1.4.0          digest_0.6.29
## [127] sctransform_0.3.3      RcppAnnoy_0.0.19       spatstat.data_2.1-2
## [130] rmarkdown_2.11         leiden_0.3.9           htmlTable_2.4.0
## [133] uwot_0.1.11            shiny_1.7.1            lifecycle_1.0.1
```

```
## [136] nlme_3.1-155          jsonlite_1.7.3        viridisLite_0.4.0     75
## [139] limma_3.46.0          fansi_1.0.2           pillar_1.7.0          76
## [142] lattice_0.20-41       ggrastr_1.0.1         fastmap_1.1.0         77
## [145] httr_1.4.2            survival_3.2-7        glue_1.6.1            78
## [148] fdrtool_1.2.17        png_0.1-7             iterators_1.0.14      79
## [151] bit_4.0.4             class_7.3-17          stringi_1.7.6         80
## [154] blob_1.2.2            caTools_1.18.2        latticeExtra_0.6-29   81
## [157] memoise_2.0.1         irlba_2.3.5           e1071_1.7-9           82
## [160] future.apply_1.8.1                                                83
```