

# Monocytes can Proliferate in Vacant Tissue Niches prior to Differentiation into Macrophages

6-Merge two experiments and plot presentation

2022-01-28 14:36:46 +0100

## Abstract

Resident tissue macrophages (RTM) are differentiated immune cells populating distinct niches and exhibiting important tissue-supportive functions. RTM maintenance is thought to depend either on monocyte engraftment and differentiation, or on the self-renewal of mature RTM. Here, we discovered that monocytes can re-enter cell cycle and proliferate locally before their differentiation into RTM. We developed a mouse model of inducible lung interstitial macrophage (IM) depletion in which the vacant niche is repopulated by BM-derived monocytes giving rise to fully differentiated IM subsets. By performing time-course single-cell RNA-sequencing analyses of myeloid cells during niche refilling, we found that few Ly6C+ classical monocytes could self-renew locally in a CSF1R-dependent manner. We further showed that the transcription factor MafB restricted such proliferation and was essential to mediate RTM specification and identity in our model. Our data 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

<b>1 Description</b>	<b>2</b>
<b>2 Load data and packages</b>	<b>2</b>
<b>3 Merge Exp1 and Exp2</b>	<b>2</b>
3.1 Prepare metadata: . . . . .	2
3.2 Normalize and process: . . . . .	2
3.3 Non-linear reduction: . . . . .	5
3.4 Presentation in 3D . . . . .	5
3.5 Cluster cells . . . . .	6
<b>4 Population characterization</b>	<b>8</b>
4.1 Characterization of Cluster 6 . . . . .	10
4.2 Cell-cycle analysis . . . . .	14
<b>5 Generate plots</b>	<b>16</b>
<b>6 Session information</b>	<b>34</b>

# 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 experiments (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:

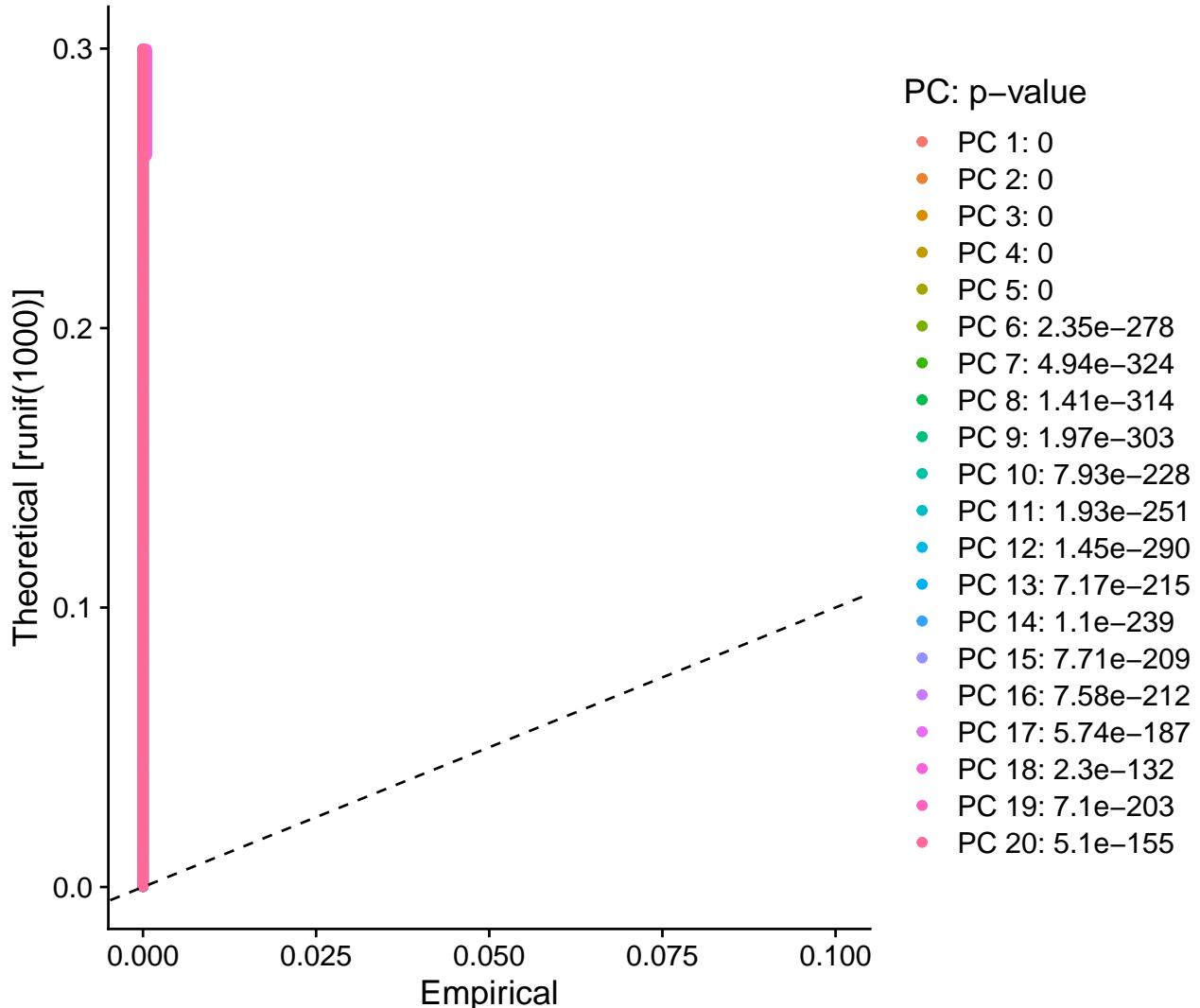
```
imdtr3 <- merge(imdtr1, imdtr2, project = "IM-DTR")
```

### 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.

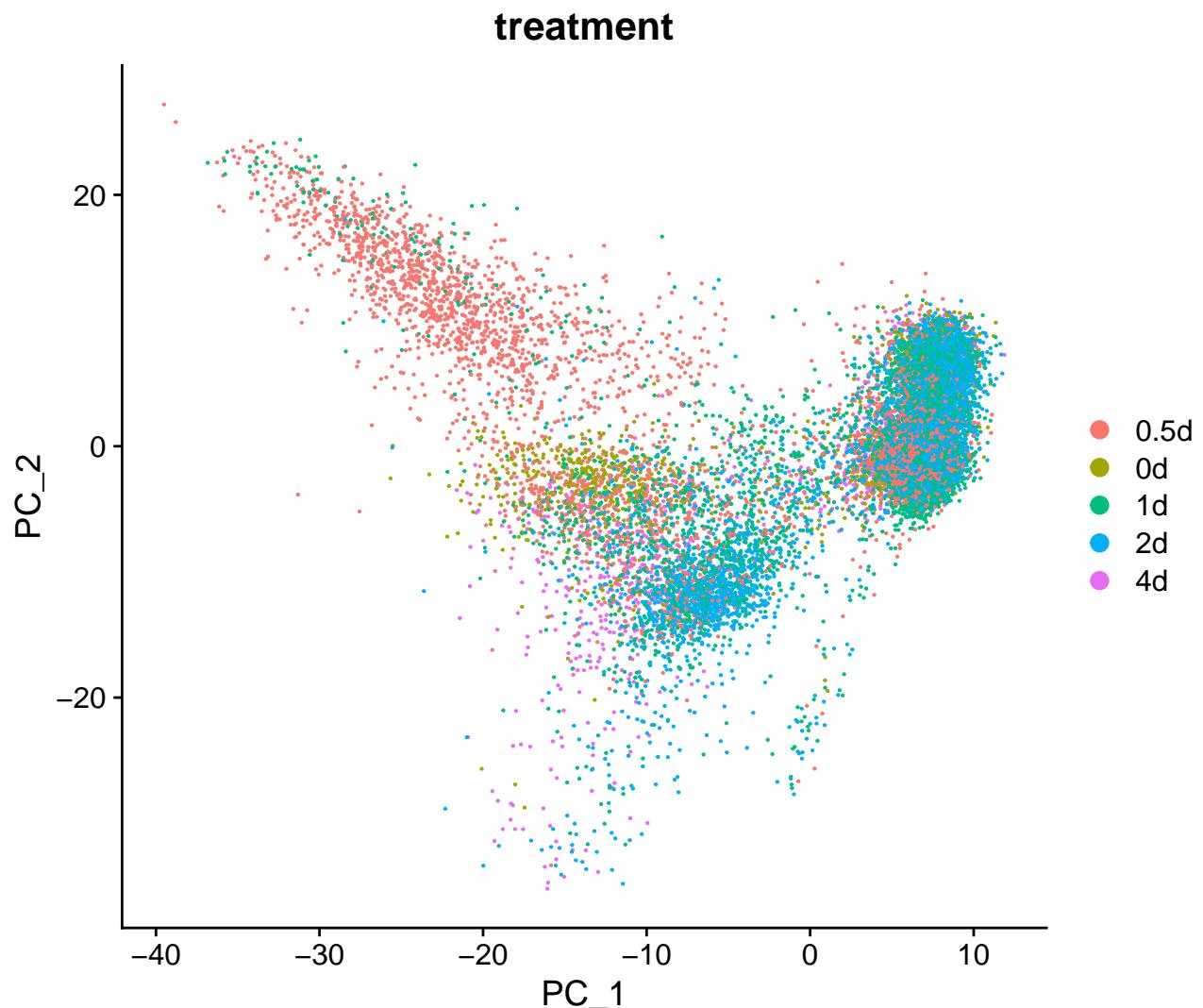
Remove old calculated neighbor (snn):

```
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))
```

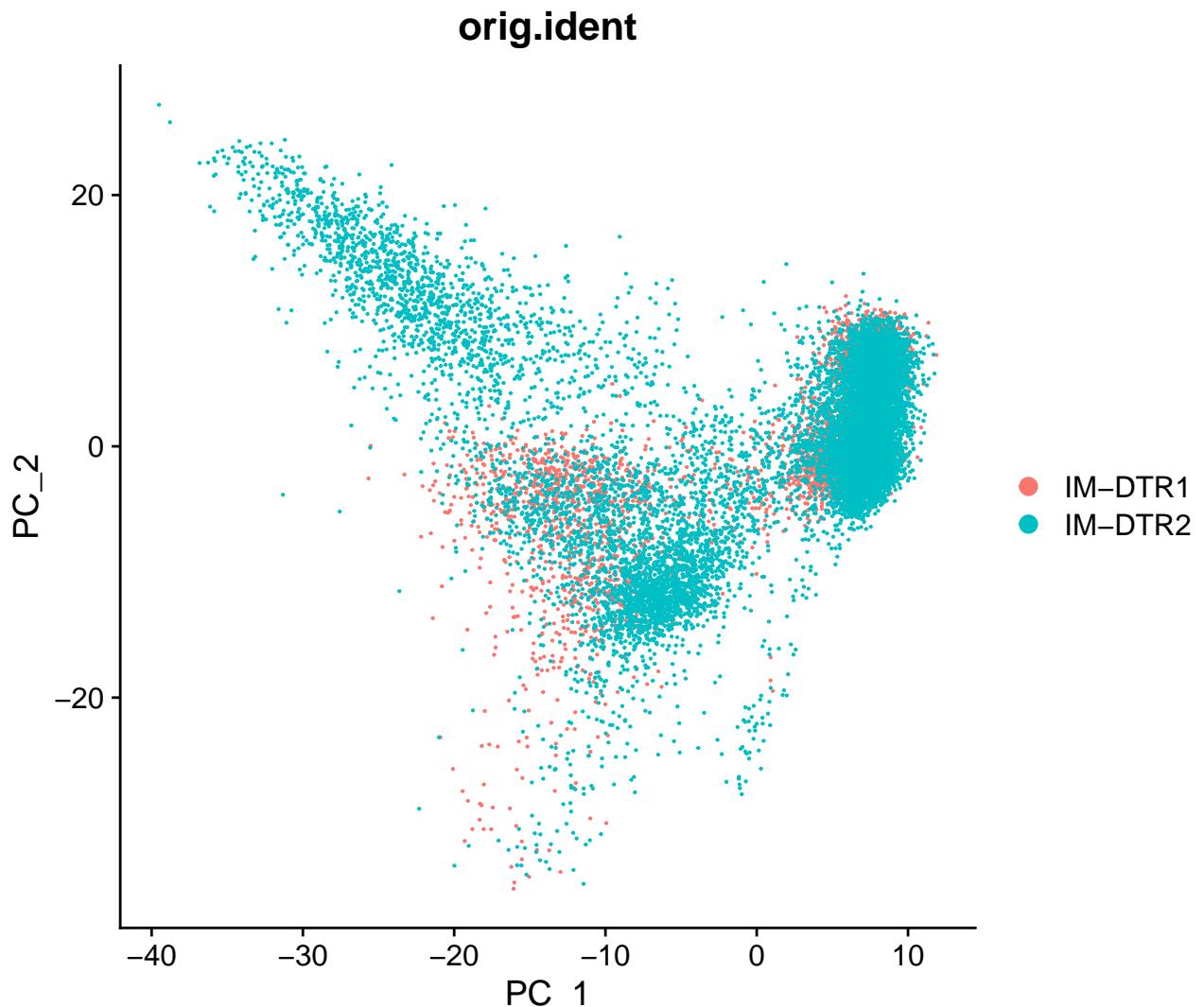
```
PCAPlot(imdtr3, group.by = "treatment")
```

1



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

1



*Samples from Exp1 and Exp2 are so similar.*

### 3.3 Non-linear reduction:

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

1

Calculate UMAP in 3D:

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

1

### 3.4 Presentation in 3D

```
library(rgl)
library(RColorBrewer)

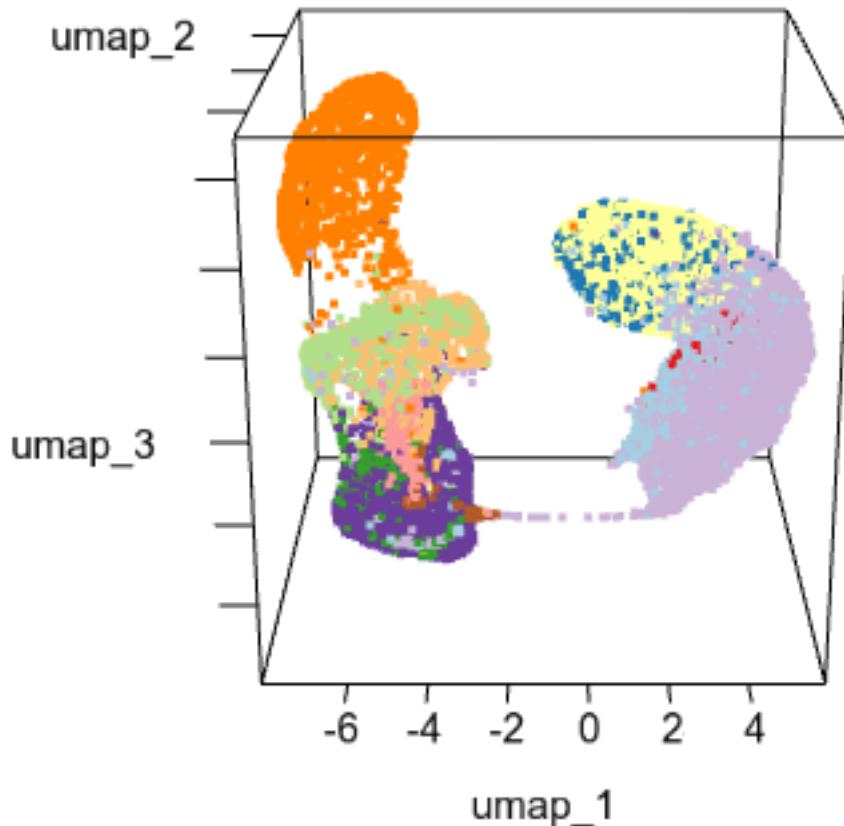
umap_1 <- imdtr3.for3d@reductions$umap@cell.embeddings[,1]
umap_2 <- imdtr3.for3d@reductions$umap@cell.embeddings[,2]
umap_3 <- imdtr3.for3d@reductions$umap@cell.embeddings[,3]
```

1  
2  
3  
4  
5  
6  
7

```

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()                                     8
9

```



*3d plot shows only one bridge exists.*

### 3.5 Cluster cells

```

imdtr3.for3d <- FindNeighbors(imdtr3.for3d, dims = 1:10)           1
imdtr3.for3d <- FindClusters(imdtr3.for3d, resolution = 0.23)        2

```

```

## Modularity Optimizer version 1.3.0 by Ludo Waltman and Nees Jan van Eck 1
##                                         2
## Number of nodes: 15941               3
## Number of edges: 508837               4
##                                         5

```

```

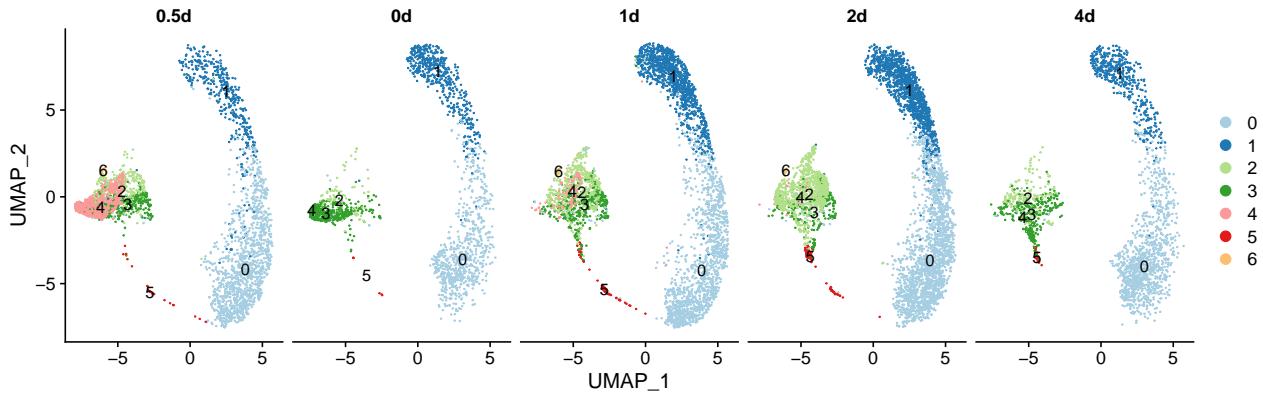
## Running Louvain algorithm...
## Maximum modularity in 10 random starts: 0.9276
## Number of communities: 7
## Elapsed time: 1 seconds

```

```

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

```

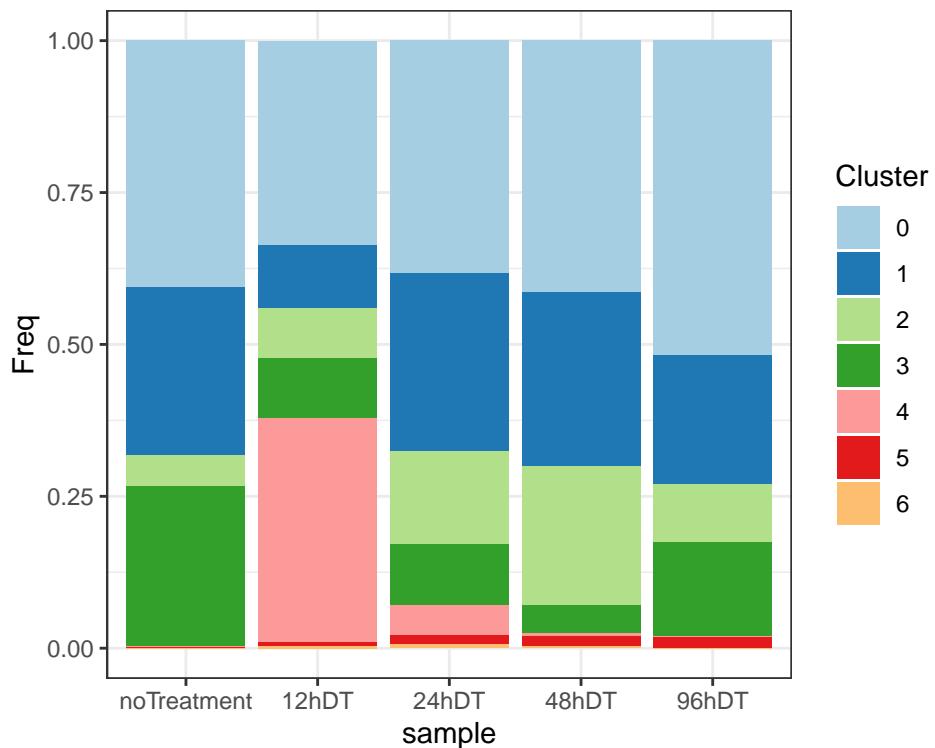


*The cluster 6 could be DCs*

```

1 source("../R/SeuratFreqTable.R")
2 freq.celltype.list <- list(
3   `noTreatment` = Seurat2CellFreqTable(subset(imdtr3.for3d, subset =
4     treatment == "0d"), slotName = "RNA_snn_res.0.23"),
5   `12hDT` = Seurat2CellFreqTable(subset(imdtr3.for3d, subset = treatment
6     == "0.5d"), slotName = "RNA_snn_res.0.23"),
7   `24hDT` = Seurat2CellFreqTable(subset(imdtr3.for3d, subset = treatment
8     == "1d"), slotName = "RNA_snn_res.0.23"),
9   `48hDT` = Seurat2CellFreqTable(subset(imdtr3.for3d, subset = treatment
10    == "2d"), slotName = "RNA_snn_res.0.23"),
11   `96hDT` = Seurat2CellFreqTable(subset(imdtr3.for3d, subset = treatment
12    == "4d"), slotName = "RNA_snn_res.0.23")
13 )
14
15 source("../R/barChart.R")
16 barChart(freq.celltype.list) + labs(fill = "Cluster") + scale_fill_manual(
17   values = pal)

```

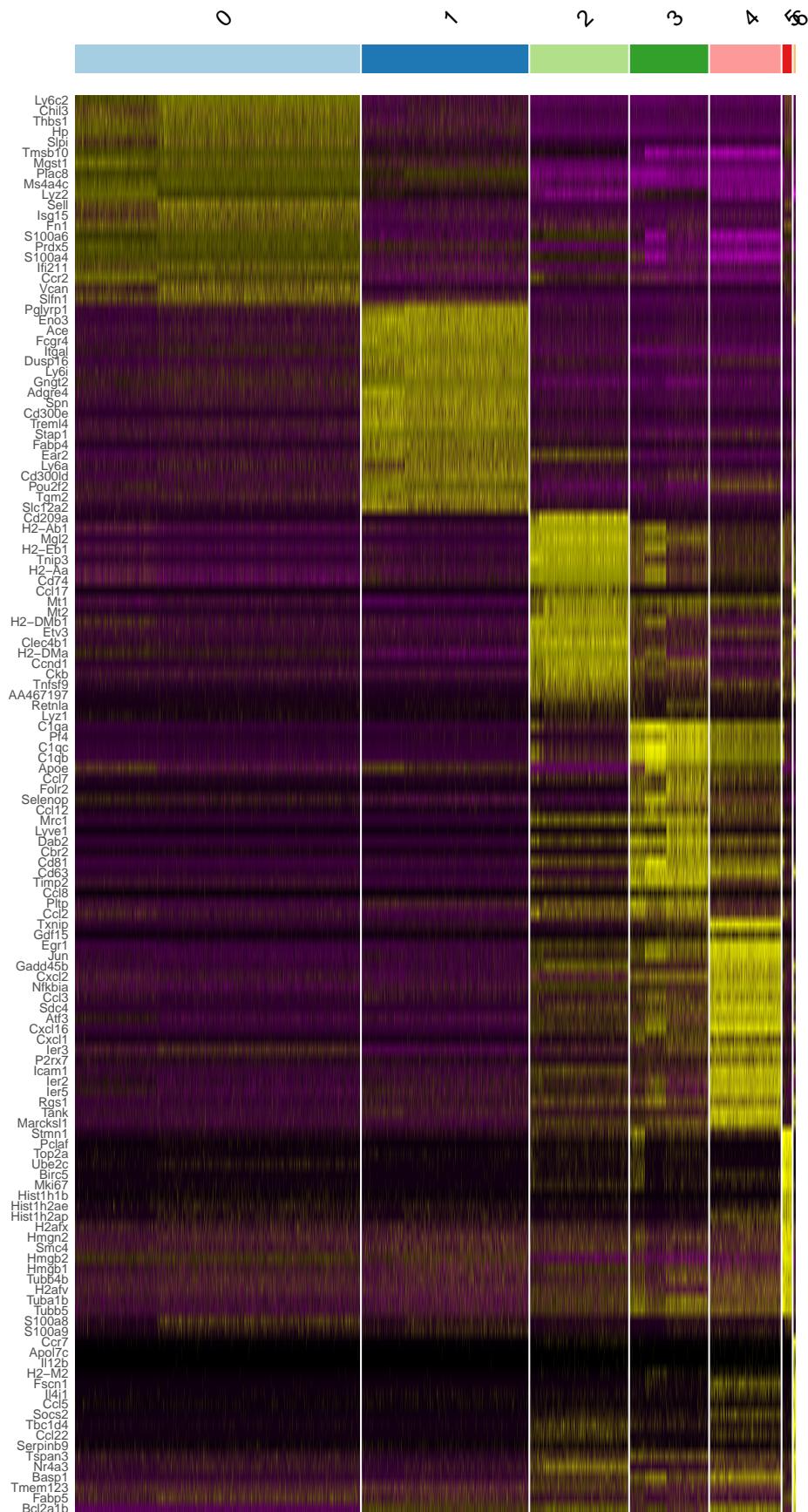


## 4 Population characterization

```

library(dplyr)                                     1
all_cluster.markers <- FindAllMarkers(imdtr3.for3d, verbose = FALSE)    2
top20 <- all_cluster.markers %>% group_by(cluster) %>% top_n(n = 20, wt =
  avg_log2FC)                                         3
DoHeatmap(imdtr3.for3d, features = top20$gene, group.colors = pal) +
  NoLegend()                                           4

```



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

sampleG0data <- new("topGOdata",
  description = "Simple_session", ontology = "BP",
  allGenes = geneList,
  annot = annFUN.org, mapping = "org.Mm.eg.db", ID =
  "SYMBOL")
resultFisher <- runTest(sampleG0data, algorithm = "classic", statistic = "
  fisher")
GenTable(sampleG0data, classicFisher = resultFisher, topNodes = 20)

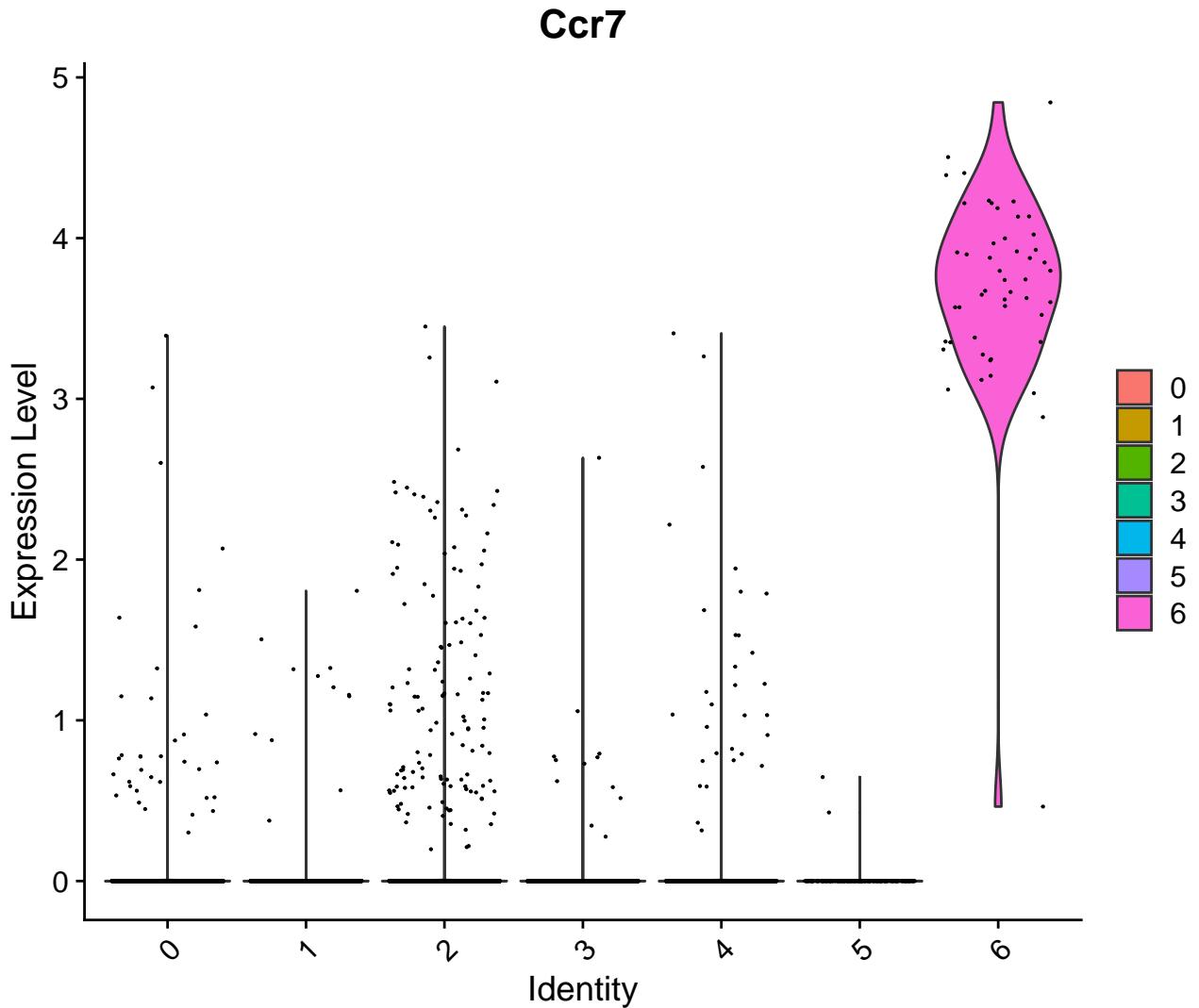
```

	## # A tibble: 20 x 6	## GO.ID	Term	Annotated	Significant	Expected
	## <chr>	## <chr>		<int>	<int>	<dbl> <chr>
1	## <-->	## >				
2	## 1	## GO:0048523	negative regulation ~	4950	173	77.2 2.4e
3	## 2	## GO:0048519	negative regulation ~	5500	181	85.8 8.5e
4	## 3	## GO:0009987	cellular process	17114	343	267. 2.6e
5	## 4	## GO:0048518	positive regulation ~	6358	189	99.2 9.8e
6	## 5	## GO:0065009	regulation of molecu~	2574	109	40.2 3.6e
7	## 6	## GO:0050789	regulation of biolog~	12212	280	191. 1.6e
8	## 7	## GO:0050790	regulation of cataly~	1876	89	29.3 7.2e
9	## 8	## GO:0048522	positive regulation ~	5834	175	91.0 9.1e
10	## 9	## GO:0050794	regulation of cellul~	11652	270	182. 1.6e
11	## 10	## GO:0002376	immune system process	2832	112	44.2 1.8e
12			-21			
13			-21			

## 11 GO:0065007 biological regulation -21	12806	286	200.	1.9e	14
## 12 GO:0040011 locomotion -21	1863	87	29.1	6.6e	15
## 13 GO:0016477 cell migration -21	1501	77	23.4	8.3e	16
## 14 GO:0048870 cell motility -20	1663	81	26.0	1.6e	17
## 15 GO:0051674 localization of cell -20	1663	81	26.0	1.6e	18
## 16 GO:0048856 anatomical structure~ -20	5977	174	93.3	3.9e	19
## 17 GO:0051336 regulation of hydrol~ -20	1002	61	15.6	4.7e	20
## 18 GO:0032502 developmental process -20	6449	182	101.	7.7e	21
## 19 GO:0006928 movement of cell or ~ -19	2101	91	32.8	1.0e	22
## 20 GO:0048583 regulation of respon~ -19	3972	133	62.0	1.9e	23

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

1



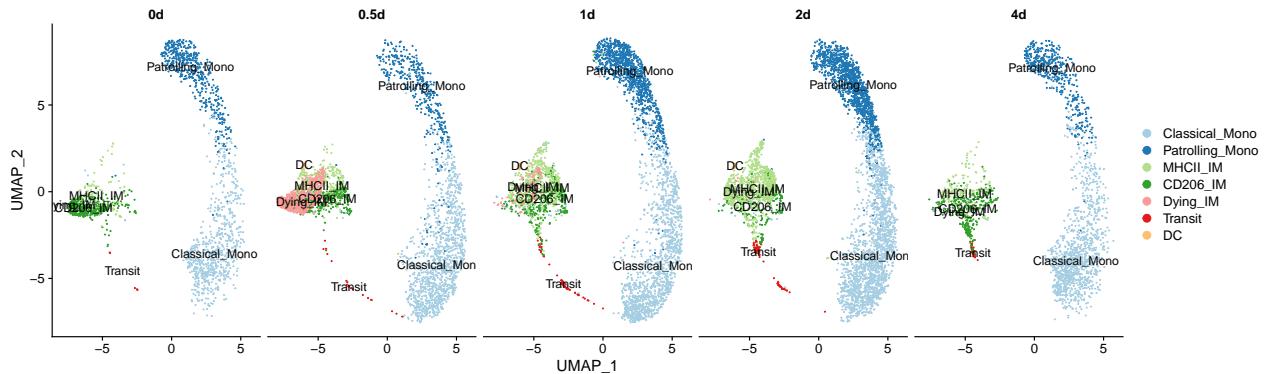
Cluster 0: ClaMo Cluster 1: PatMo Cluster 2: MHCII\_IM Cluster 3: CD206\_IM Cluster 4: Dying\_IM  
Cluster 5: Transit Cluster 6: DC

change the idents

```

levels(imdtr3.for3d) 1
## [1] "0" "1" "2" "3" "4" "5" "6" 1
imdtr3.for3d$cell.type2 <- factor(Idents(imdtr3.for3d), labels = c(" 1
  Classical_Mono", "Patrolling_Mono", "MHCII_IM", "CD206_IM", "Dying_IM",
  "Transit", "DC")) 2
3
Idents(imdtr3.for3d) <- "cell.type2" 4
imdtr3.for3d$treatment <- factor(imdtr3.for3d$treatment, levels = c("0d", 5
  "0.5d", "1d", "2d", "4d"))
DimPlot(imdtr3.for3d, label = TRUE, split.by = "treatment", cols = pal) 6

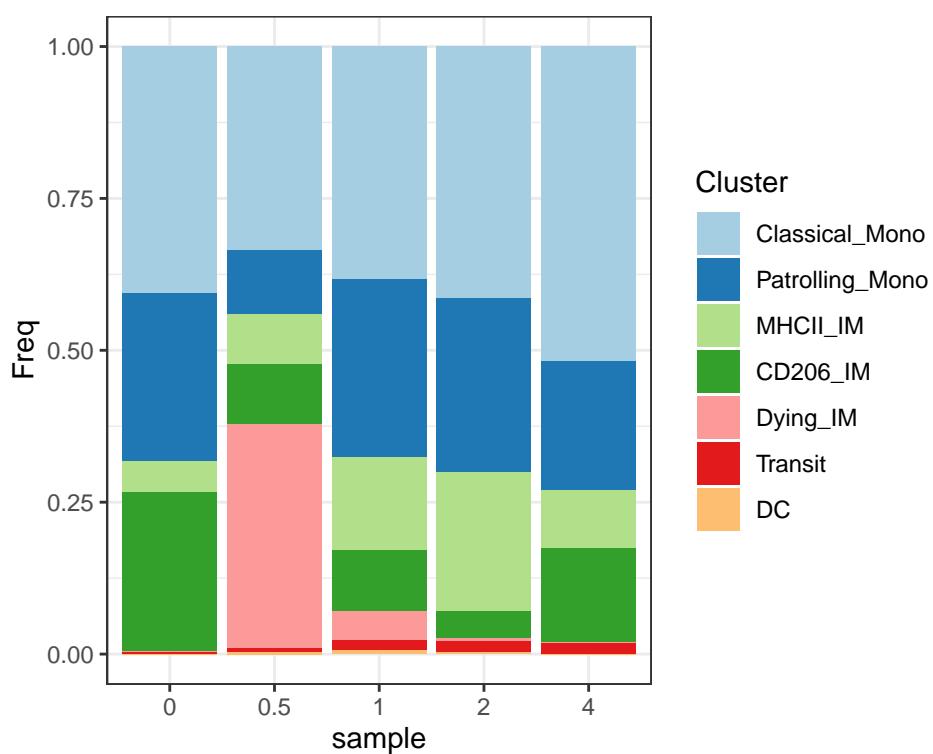
```



```

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", 1
  
```

```

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

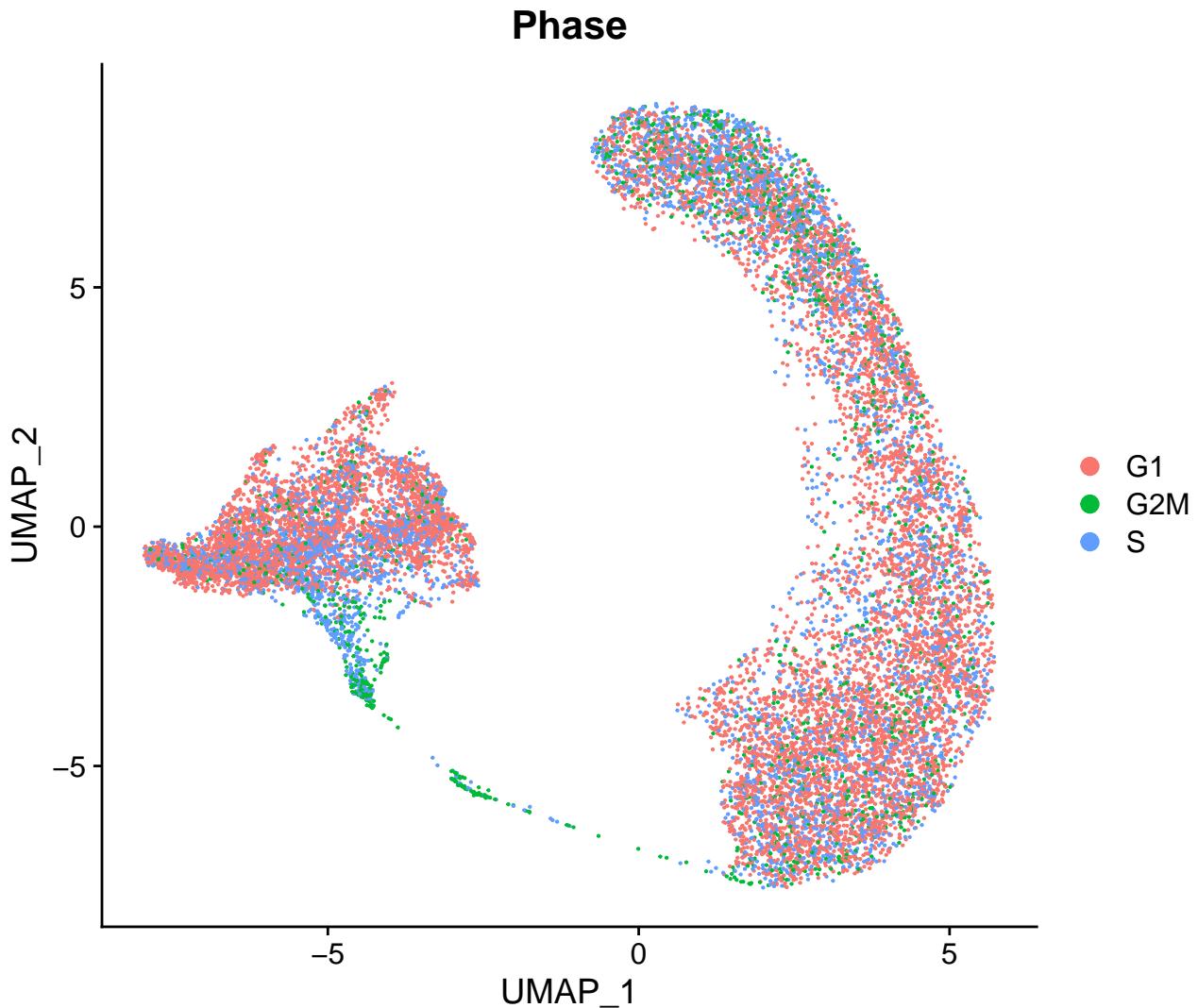
```

## 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")

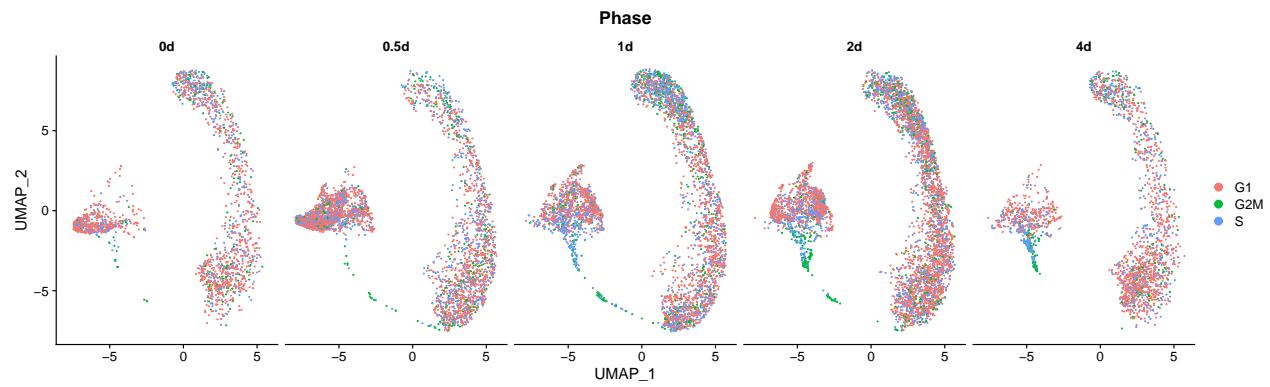
```



```

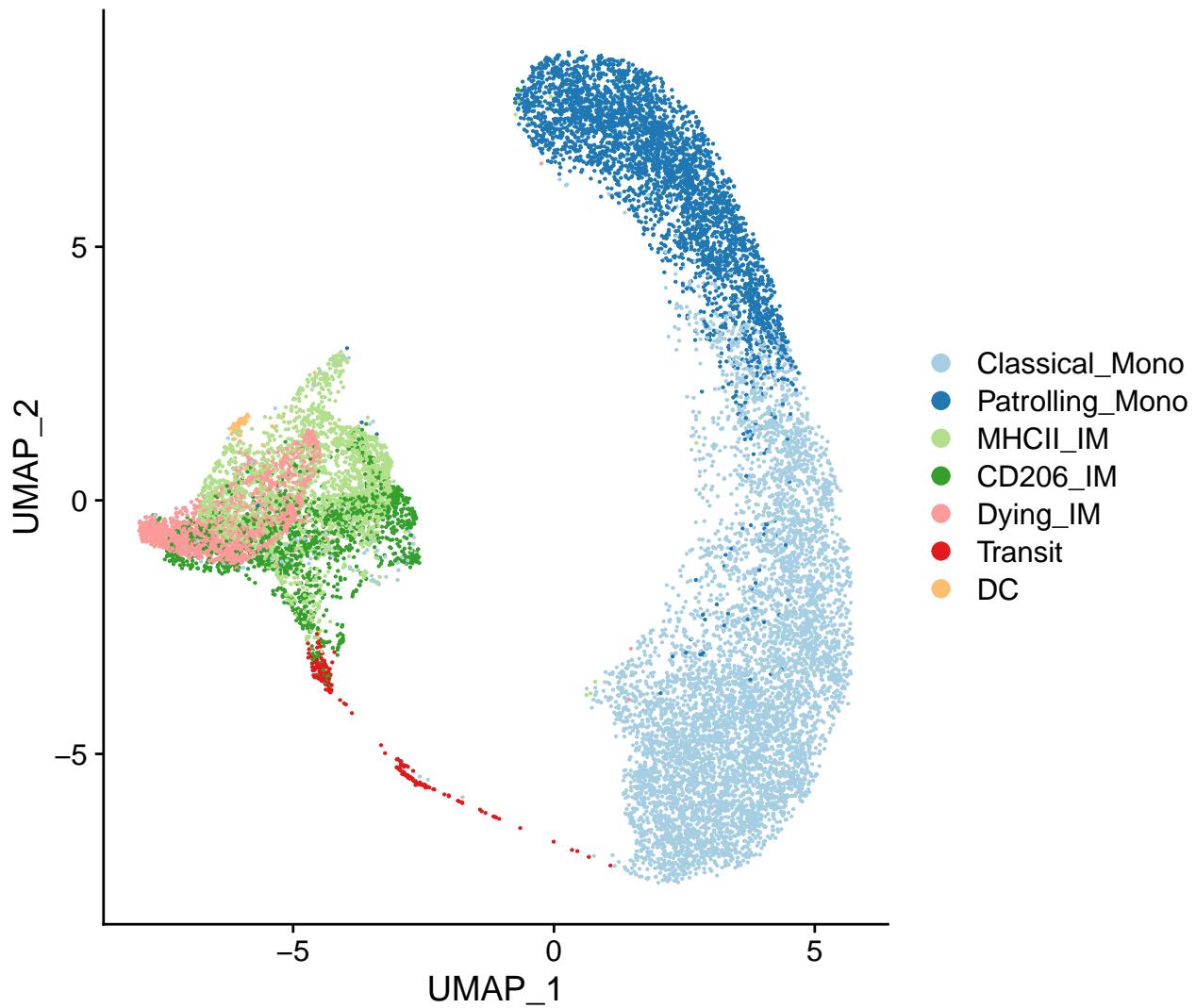
DimPlot(imdtr3.for3d, group.by = "Phase", reduction = "umap", split.by = "
  treatment")

```



2d plot:

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

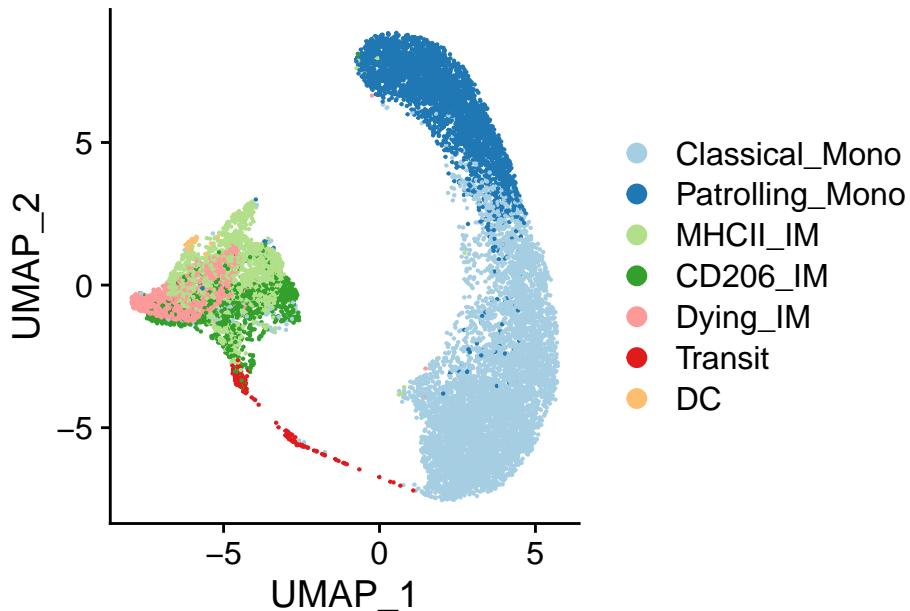


```
saveRDS(imdtr3.for3d, file = "./immune_imdtr3.seuratObject.Rds")
```

## 5 Generate plots

2D plot (component 1 and 2):

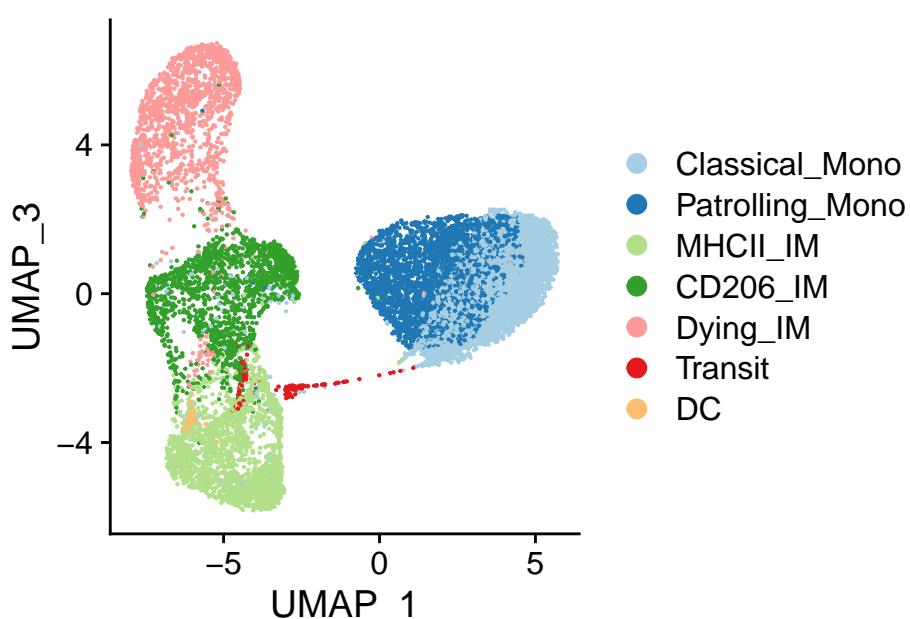
```
DimPlot(imdtr3.for3d, cols = pal, dims = c(1,2))
```



```
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))
```



```

1 ggsave(filename = "UMAPplot_All_cells_PC1_3.pdf",
2        path = "../Figures",
3        height = 3.5, width = 5,
4        device = "pdf")

```

3D plot:

```

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

```

```
open3d()
```

```

1 ## glX
2 ## 5

```

```

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

```



```
#close3d()
```

1

```
open3d()
```

1

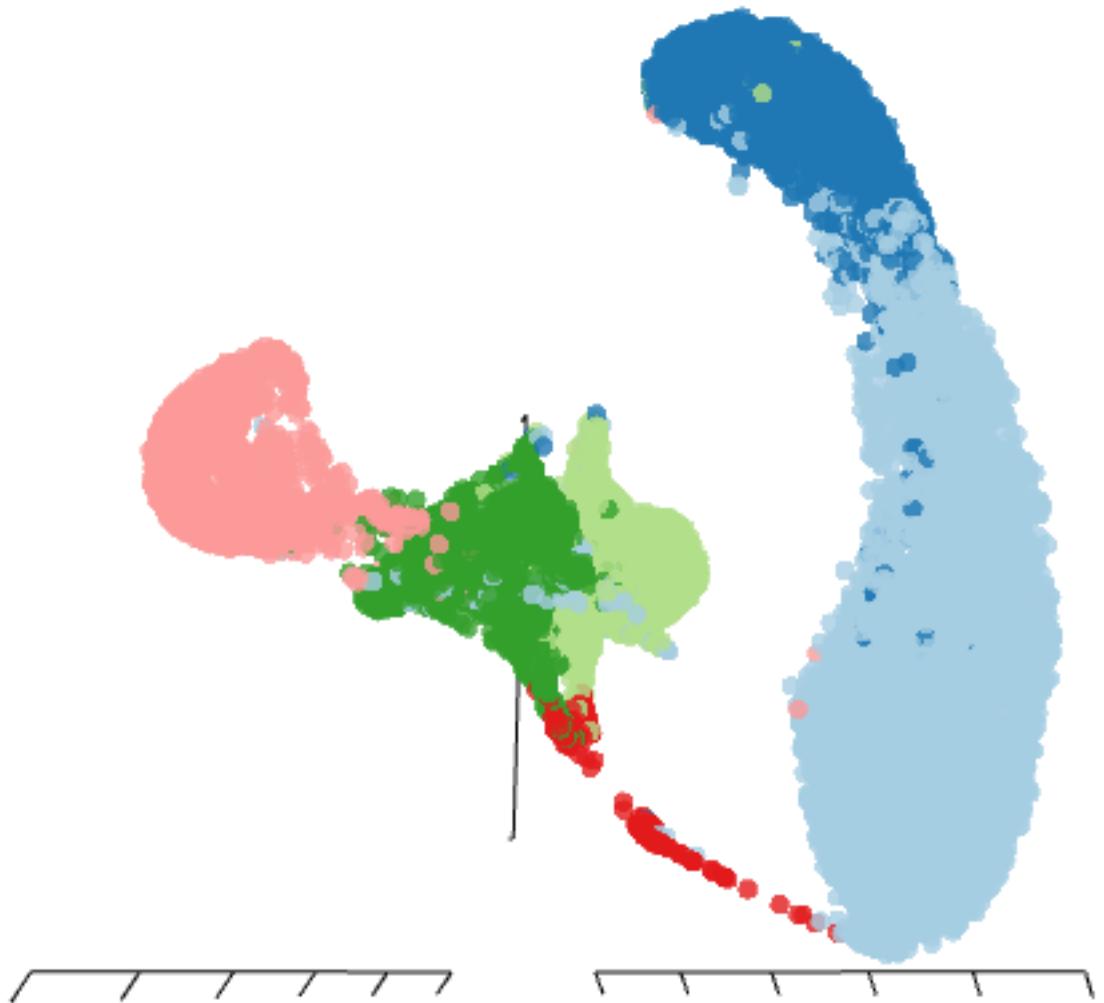
```
## glX  
## 8
```

1  
2

```
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)
```

1  
2  
3  
4  
5  
6  
7

```
| #title3d(xlab = 'UMAP1', ylab = 'UMAP2', zlab = 'UMAP3', alpha = 0.7)          8
| rgl.bringtotopt()                  9
| rgl.viewpoint(30, -20, zoom = 0.7)      10
| # rgl.postscript(filename = "../Docs/Figures/test.eps", fmt = "eps",        11
|   drawText = TRUE )
| rglwidget()                         12
```

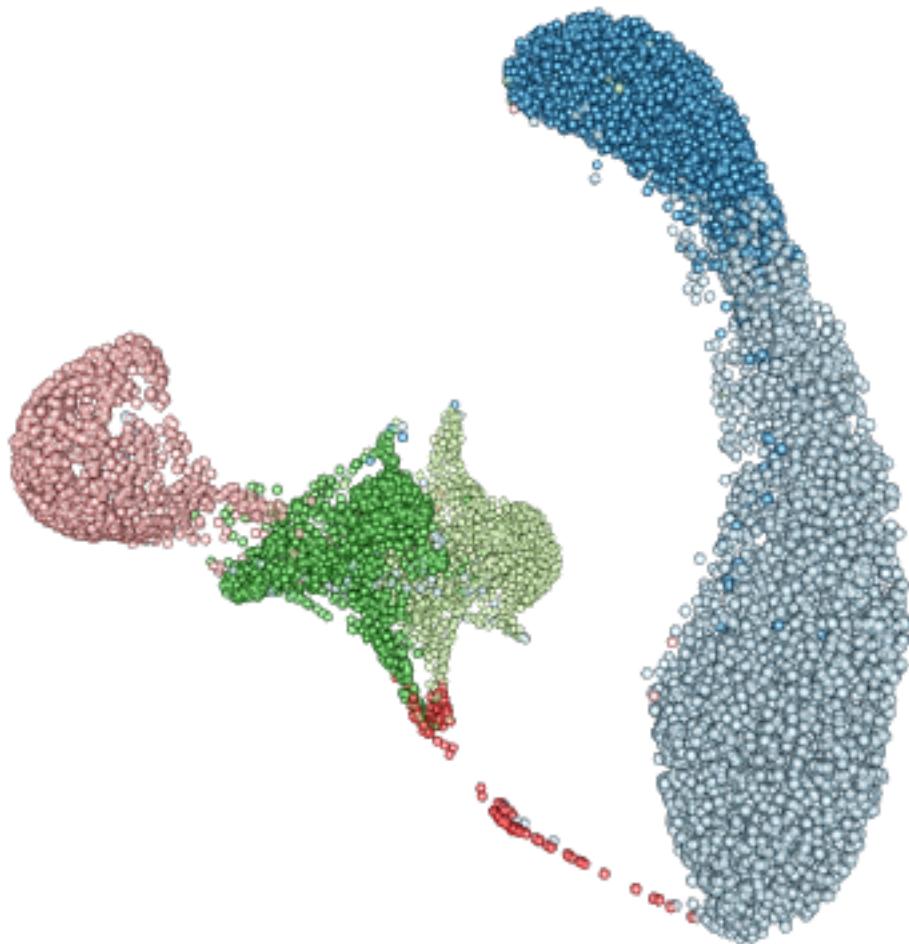


```
| #close3d()                           1
| open3d()                            1
| ## g1X                                1
| ## 11                                 2
| | spheres3d(x = umap_1, y = umap_2, z = umap_3,                      1
```

```

1 col = factor(Idents(imdtr3.for3d), labels = brewer.pal(length(
2   unique(imdtr3.for3d$cell.type2)), name = "Paired")),
3 radius = 0.1, alpha = 0.5,
4 # ambient      = "gray",
5 # emission     = "gray",
6 specular      = "gray",
7 shininess     = 10)
8 #decorate3d( box = FALSE, axes = FALSE, xlab = NULL, ylab = NULL, zlab =
9   NULL)
10 #axes3d(edges = c("x--", "y--", "z--"), alpha = 0.5)
11 #bbox3d(box=TRUE)
12 #title3d(xlab = 'UMAP1', ylab = 'UMAP2', zlab = 'UMAP3', alpha = 0.7)
13 rgl.bringtotopt()
14 rgl.viewpoint(30, -20, zoom = 0.7)
15 # rgl.postscript( filename = "../Docs/Figures/test.eps", fmt = "eps",
#   drawText = TRUE )
rglwidget()

```



```
#close3d()
```

1

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])
```

1

2

3

4

5

6

7

8

9

10

11

12

```

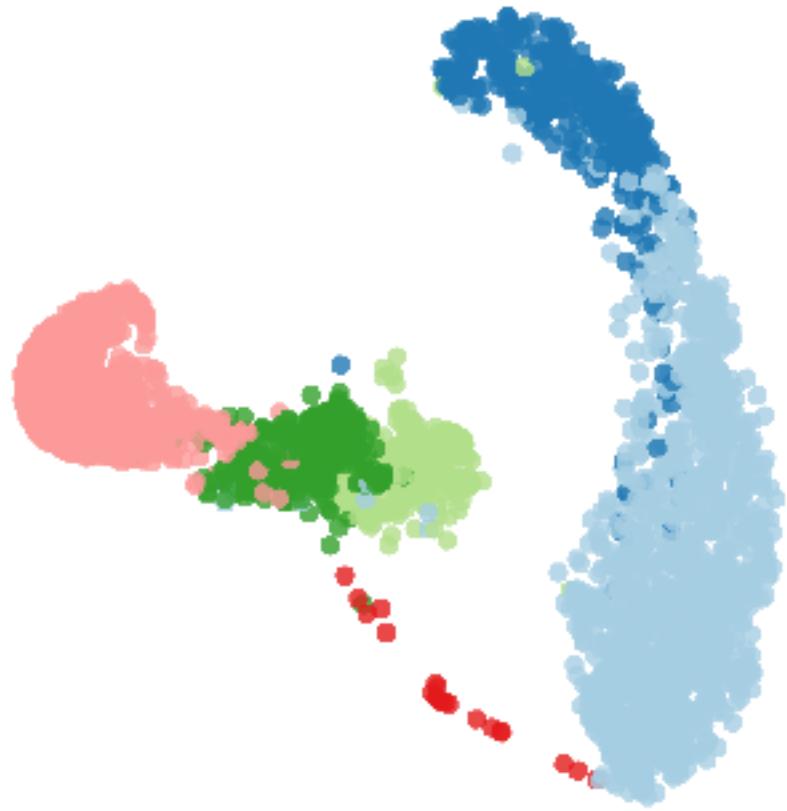
umap2.min <- min(seuratObj@reductions$umap@cell.embeddings[,2])          13
umap2.max <- max(seuratObj@reductions$umap@cell.embeddings[,2])          14
umap3.min <- min(seuratObj@reductions$umap@cell.embeddings[,3])          15
umap3.max <- max(seuratObj@reductions$umap@cell.embeddings[,3])          16
# scatterplot3d(x = umap_1, y = umap_2, z = umap_3, # old one           17
#                 color = col.pal, pch = 20,                                18
#                 grid=FALSE, box=TRUE, col.axis="black",                   19
#                 main = treat,                                         20
#                 ...)                                              21
open3d()                                                               22
points3d(x = umap_1, y = umap_2, z = umap_3,                         23
          col = col.pal,                                                 24
          size = 7.5, alpha = 0.8,                                         25
          point_antialias = TRUE,                                         26
          line_antialias = TRUE)                                         27
# decorate3d(xlim = c(umap1.min, umap1.max),                           28
#             ylim = c(umap2.min, umap2.max),                           29
#             zlim = c(umap3.min, umap3.max),                           30
#             xlab = NULL, ylab = NULL, zlab = NULL,                      31
#             box = FALSE, axes = FALSE)                                 32
# bbox3d(box=TRUE)                                                 33
#title3d(xlab = 'UMAP1', ylab = 'UMAP2', zlab = 'UMAP3', alpha = 0.7) 34
rgl.bringtotopt()                                                 35
#rgl.viewpoint(3, 180)                                              36
# rgl.postscript( filename = "../Docs/Figures/test.eps", fmt = "eps", 37
#   drawText = TRUE )
rgl.viewpoint(30, -20, zoom = 0.7)                                     38
if (savePNG) {                                                       39
  filename <- file.path("../Docs/Figures", paste(treat, ".png", sep = "")) 40
  png(filename)
  rglwidget()
  dev.off()
} else (return(rglwidget()))
#close3d()                                                       41
}

```

```

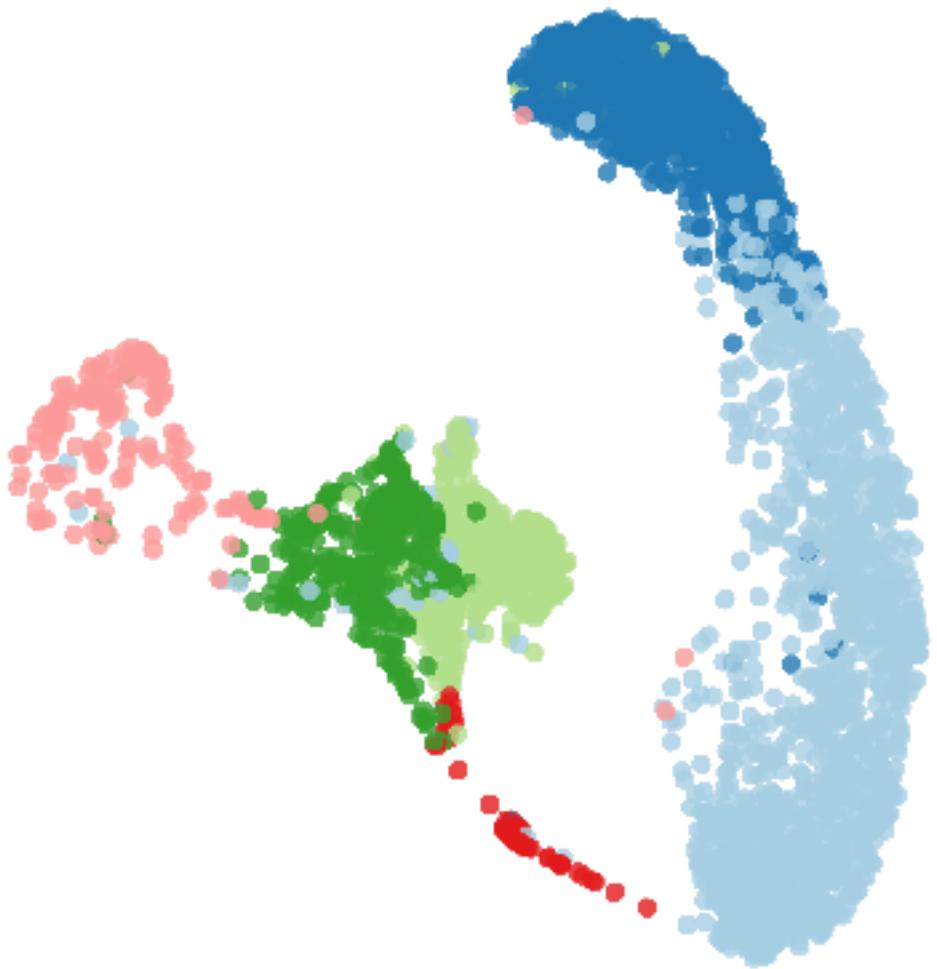
treat.list <- c("0.5d", "1d", "2d", "4d", "0d")                         1
# everytime, save PNG manually with width = 700, height = 630.            2
plot3d.byGroup(imdtr3.for3d, treat.list[1])                               3

```



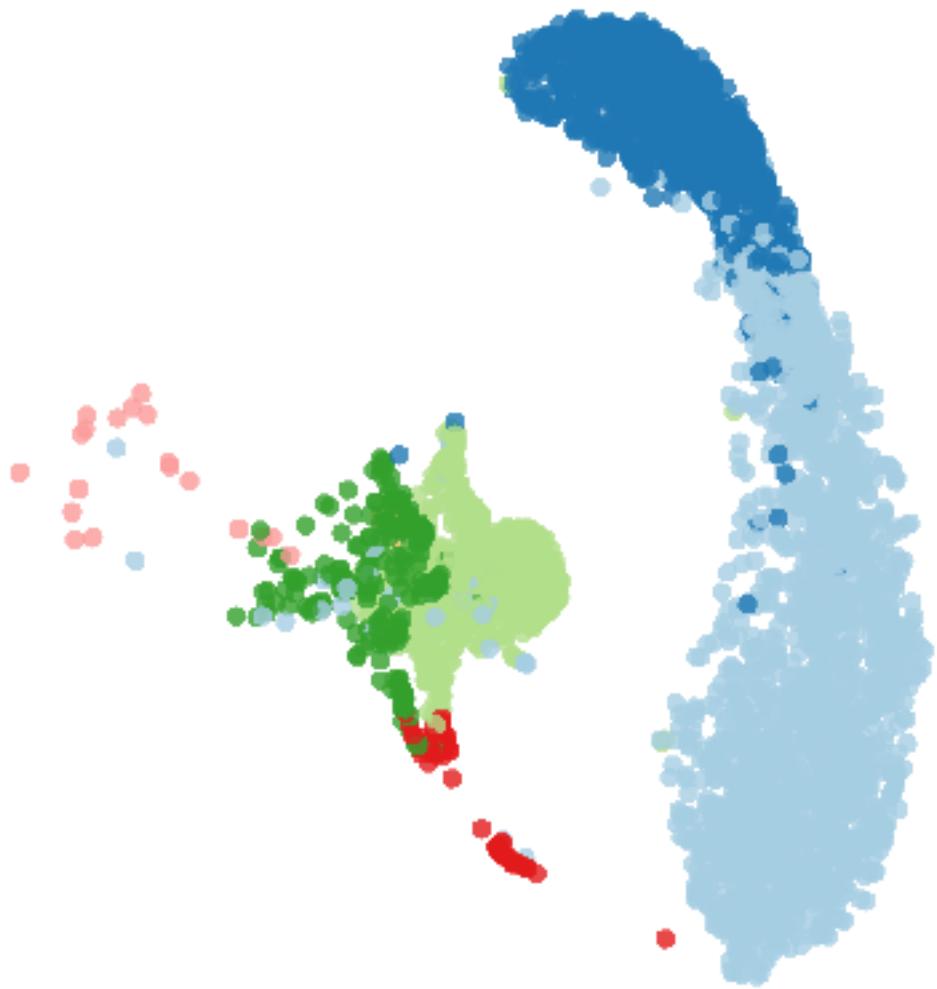
```
plot3d.byGroup(imdtr3.for3d, treat.list[2])
```

1



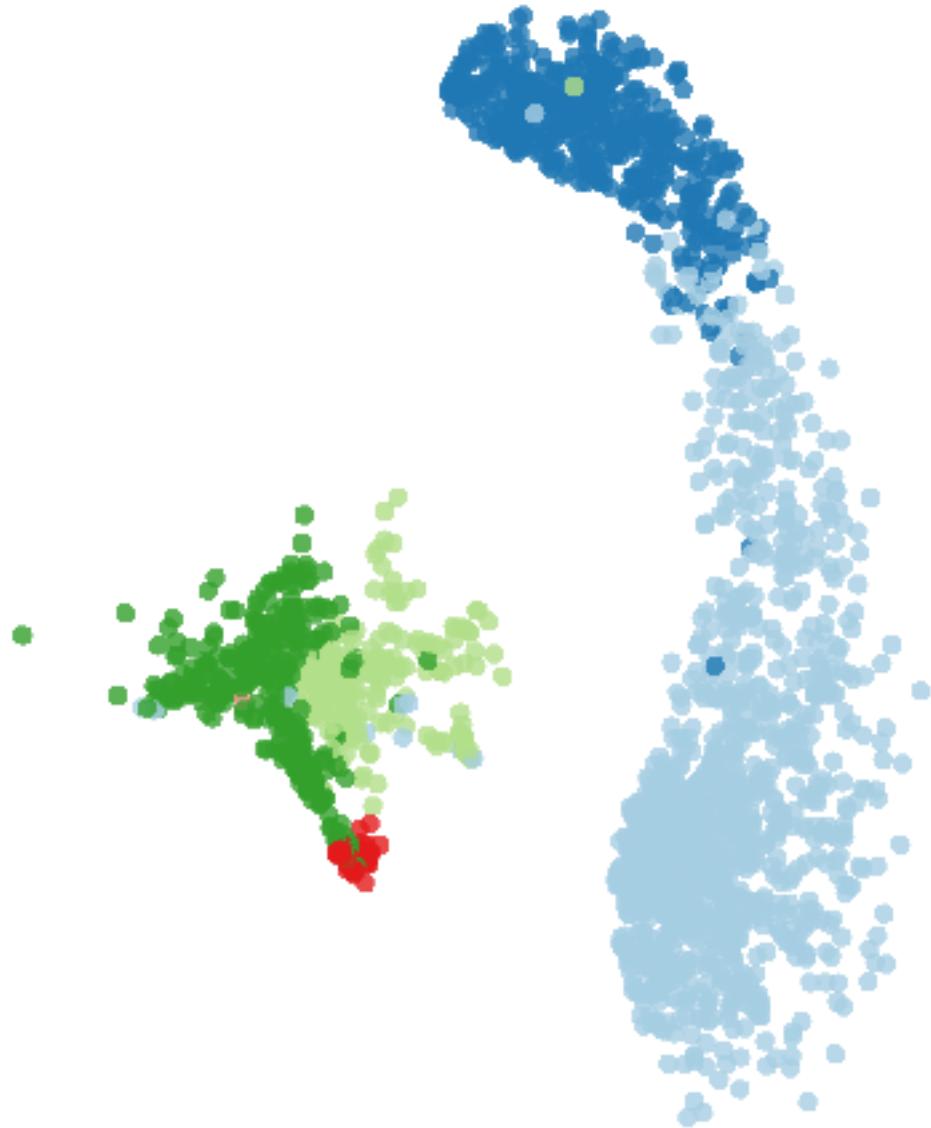
```
plot3d.byGroup(imdtr3.for3d, treat.list[3])
```

1



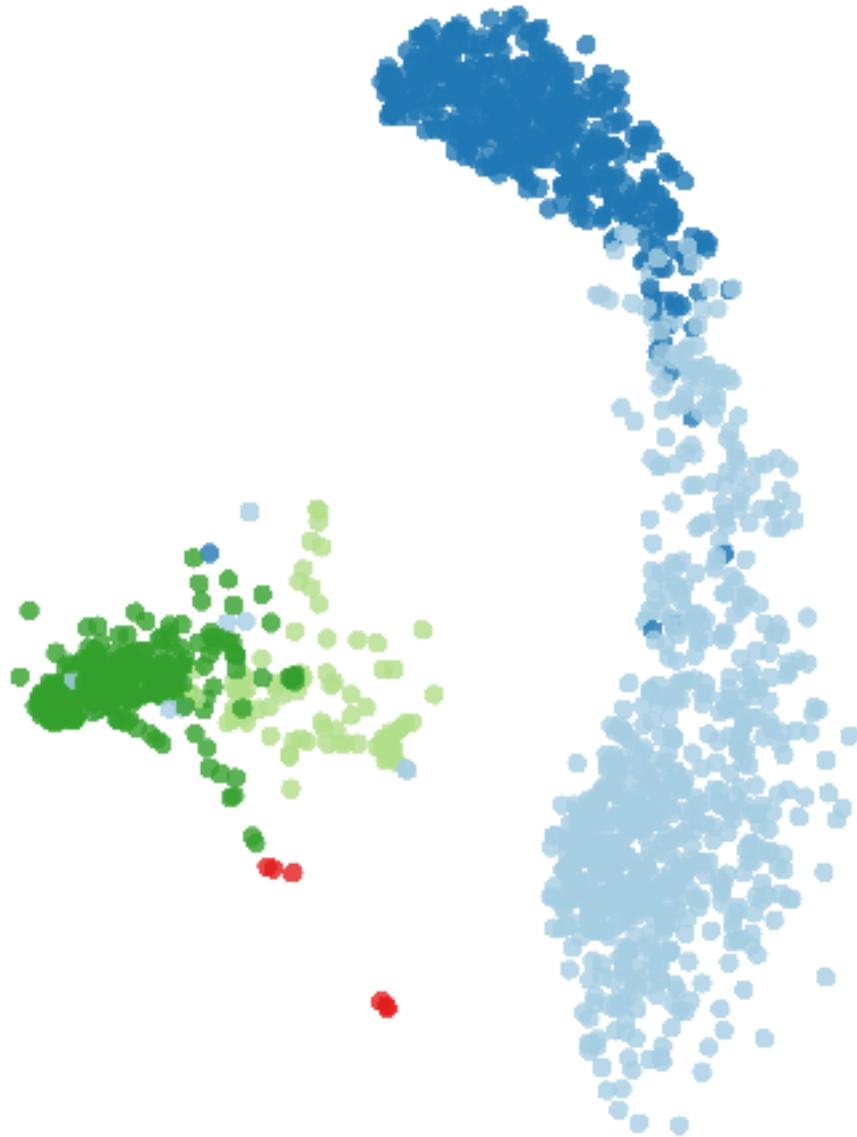
```
plot3d.byGroup(imdtr3.for3d, treat.list[4])
```

1



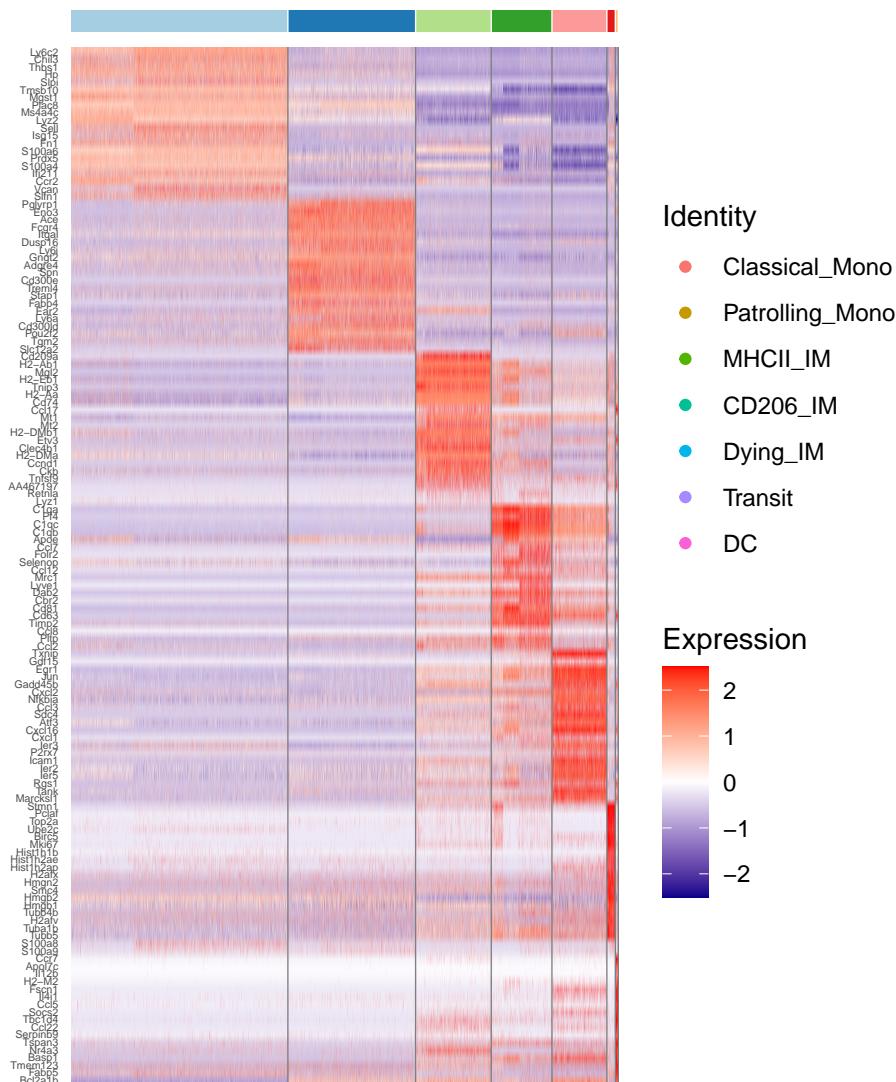
```
plot3d.byGroup(imdtr3.for3d, treat.list[5])
```

1



Heatmap:

```
DoHeatmap(imdtr3.for3d, features = top20$gene, group.colors = pal, size = 1  
0) + scale_fill_gradientn(colors = c("darkblue", "white", "red")) +  
theme(axis.text.y = element_text(size = 4)) 2
```



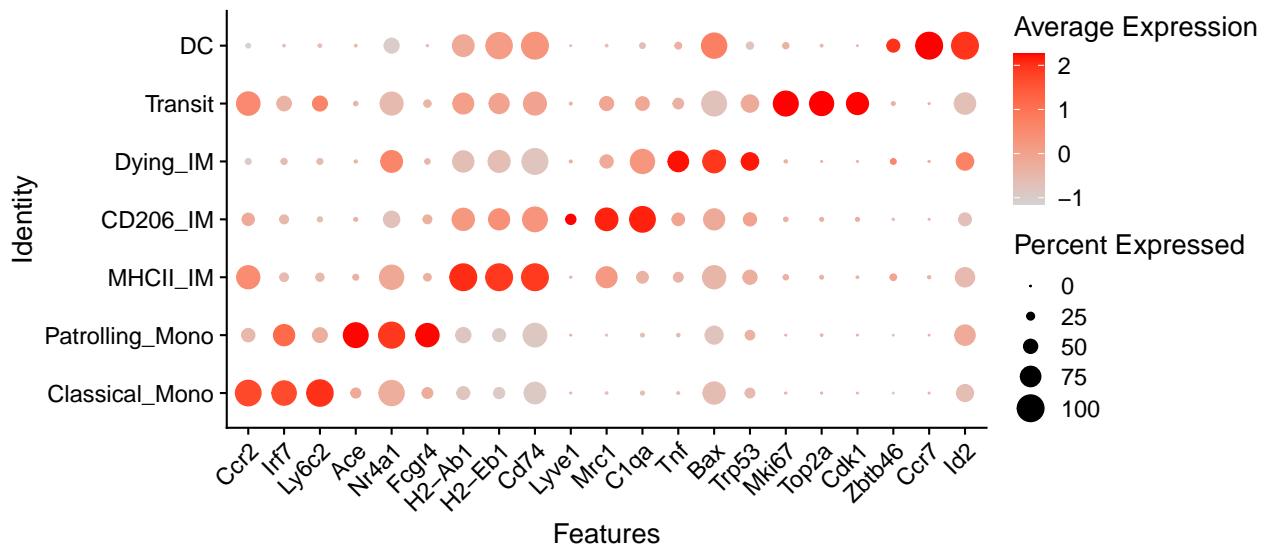
```
ggsave(filename = "HeatMap_top20_in_cell_types_blue_red.pdf",
        path = "../Figures",
        height = 6, width = 5,
        device = "pdf")
```

Dot plot show cell type markers:

```

DotPlot(imdtr3.for3d,
        cols = c("lightgrey", "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))

```



```

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

```

Cell-cycle score:

```

1 # color gradient
2 col_gradient <- colorRampPalette(c("lightgrey", "red"))(256)
3
4 # assign color in scaled cell-cycle score
5 val <- imdtr3.for3d$G2M.Score
6 col_ix <- round((val - min(val))/(max(val)-min(val)) * 255, digits = 0)
7 col_pal <- col_gradient[col_ix + 1 ] # because range will be 0:255
8 names(col_pal) <- names(col_ix)
9
10 # remake order to avoid overlay
11 umap_1.order_g2m <- umap_1[order(imdtr3.for3d$G2M.Score, decreasing = TRUE)
12   ]
13 umap_2.order_g2m <- umap_2[order(imdtr3.for3d$G2M.Score, decreasing = TRUE)
14   ]
15 umap_3.order_g2m <- umap_3[order(imdtr3.for3d$G2M.Score, decreasing = TRUE)
16   ]
17 col_pal <- col_pal[order(imdtr3.for3d$G2M.Score, decreasing = TRUE)]

```

```

1 open3d()

```

```

1 ## glX
2 ## 29

```

```

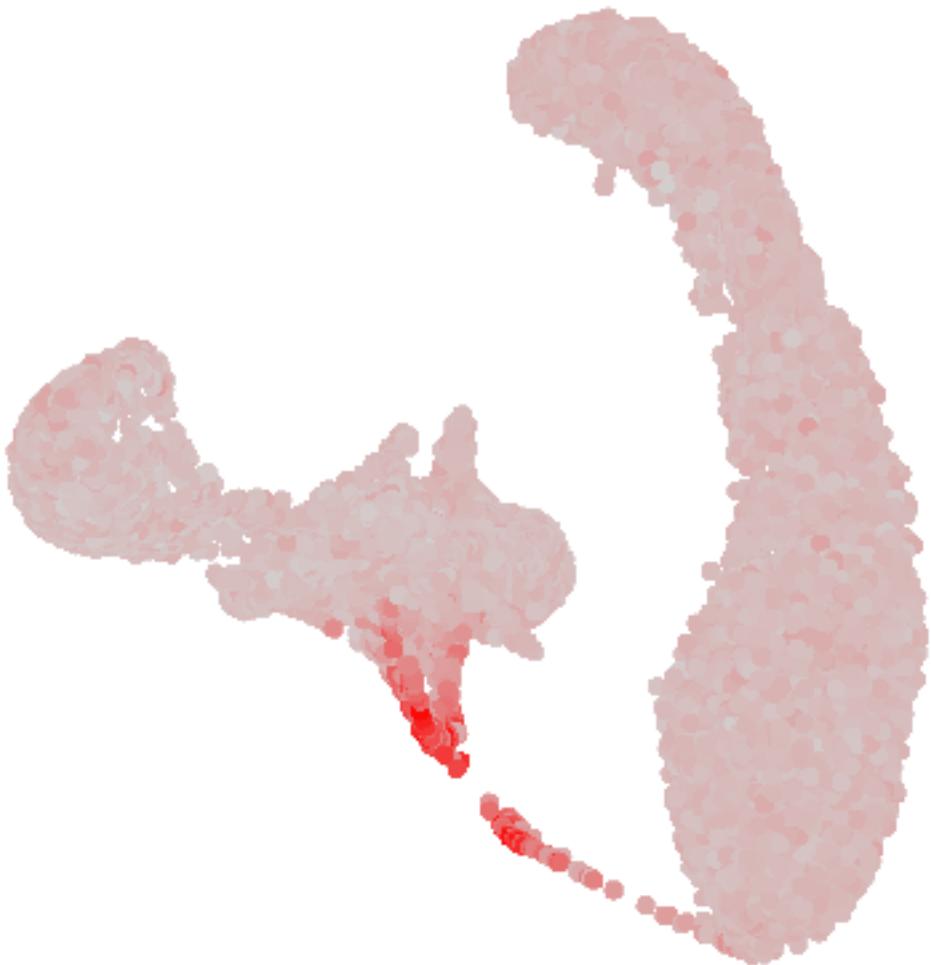
1 points3d(x = umap_1.order_g2m, y = umap_2.order_g2m, z = umap_3.order_g2m,
2           col = col_pal,
3           size = 7.5, alpha = 1,
4           point_antialias = TRUE,
5           line_antialias = TRUE)

```

```

| #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()

```



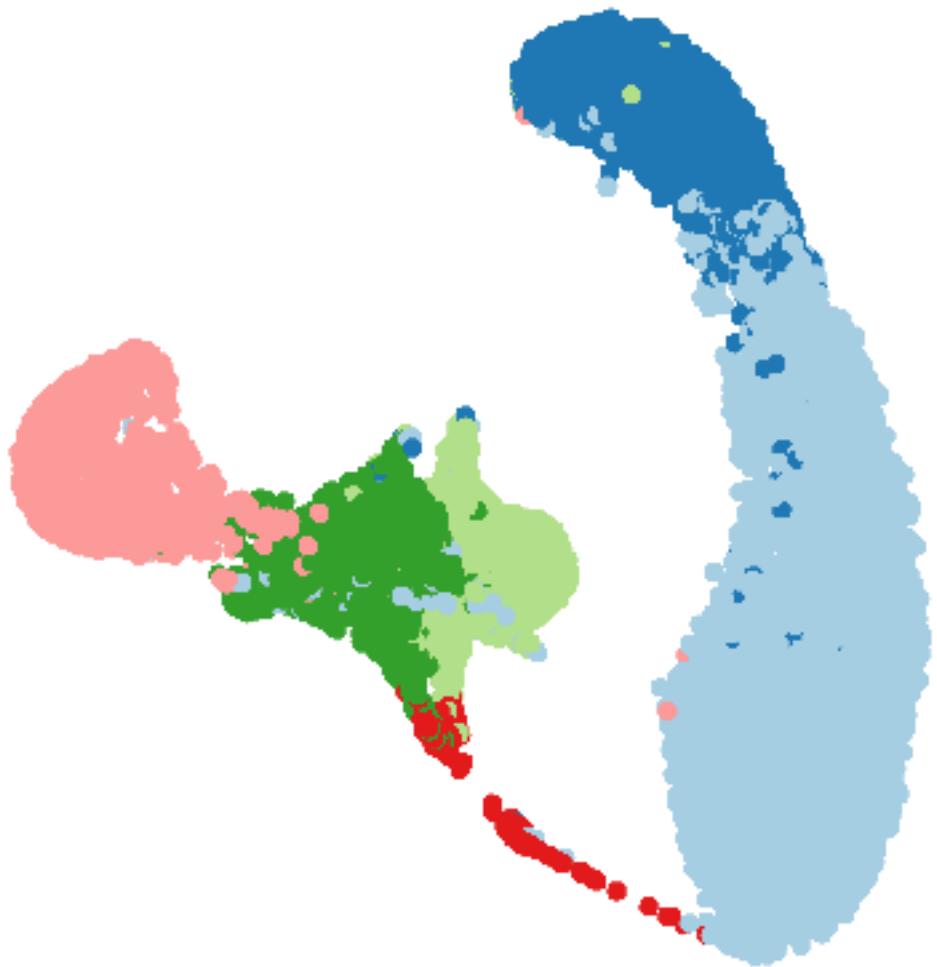
```

| #close3d() 1
|
| open3d() 1
|
| ## glX 1

```

```
## 32
```

```
points3d(x = umap_1, y = umap_2, z = umap_3,  
         col = factor(Ids(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.bringtotopt()  
rgl.viewpoint(30, -20, zoom = 0.7)  
# rgl.postscript( filename = "../Docs/Figures/test.eps", fmt = "eps",  
# drawText = TRUE )  
rglwidget()
```



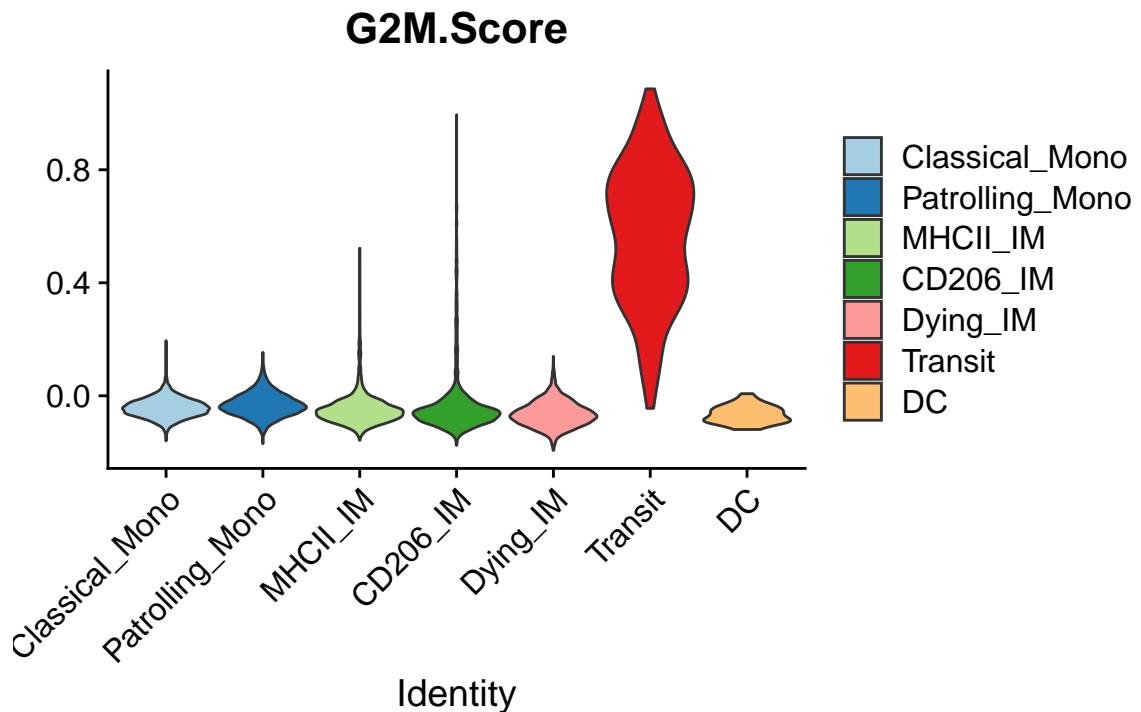
```
#close3d()
```

1

Score:

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

1



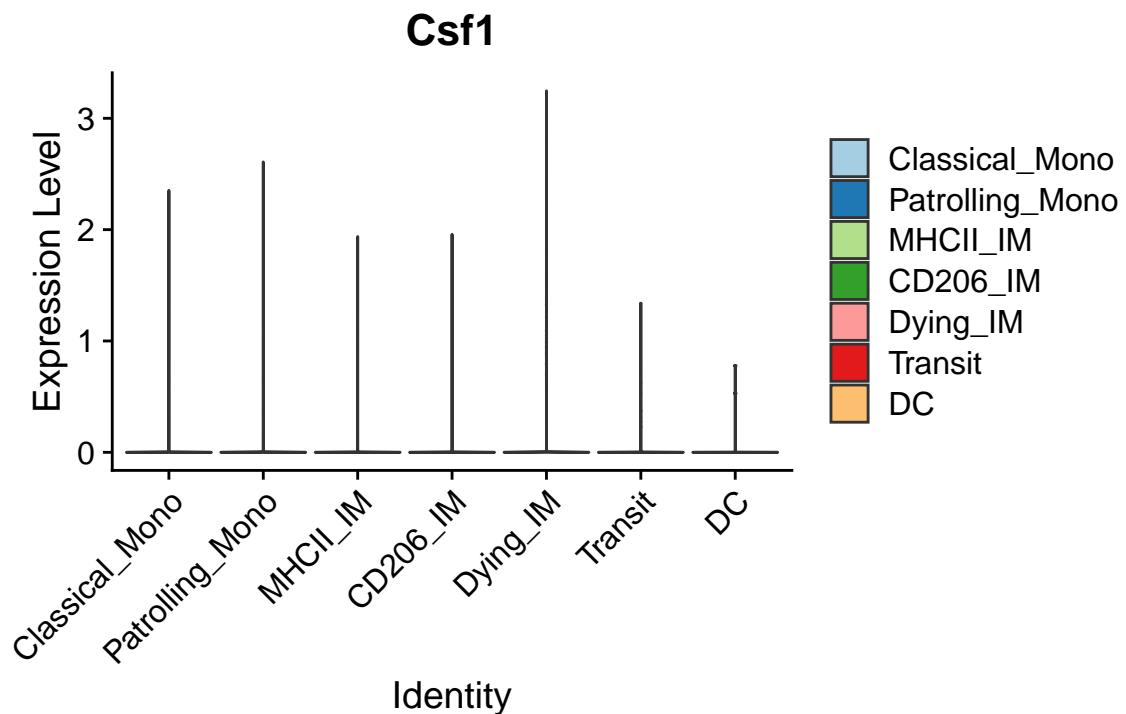
```

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

```

Csf1 expression:

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



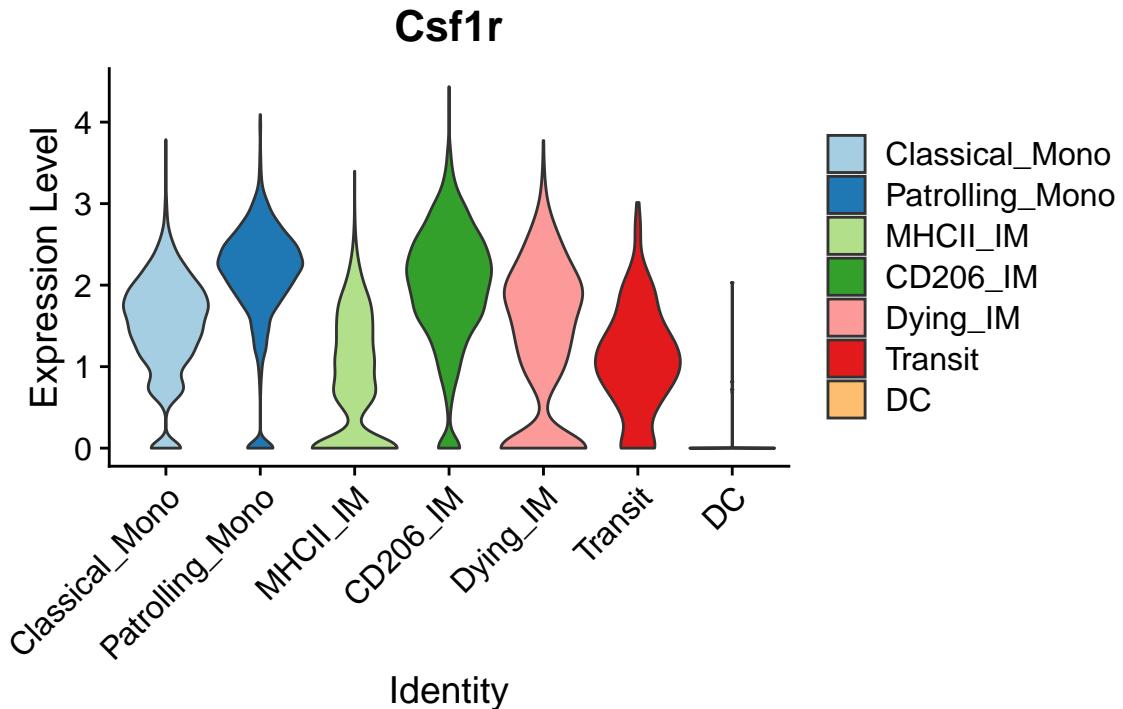
```

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

```

Csf1r

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



```

1 ggsave(filename = "VlnPlot_Csf1r_in_all_samples.pdf",
2     path = "../Figures",
3     height = 4, width = 6,
4     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	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
##			16
## attached base packages:			17
## [1] stats4 parallel stats graphics grDevices utils			18
datasets			
## [8] methods base			19
##			20
## other attached packages:			21
## [1] cowplot_1.1.1	org.Mm,eg.db_3.12.0	topGO_2.42.0	22
## [4] SparseM_1.81	GO.db_3.12.1	AnnotationDbi_1.52.0	23
## [7] IRanges_2.24.1	S4Vectors_0.28.1	Biobase_2.50.0	24
## [10] graph_1.68.0	BiocGenerics_0.36.1	dplyr_1.0.7	25
## [13] RColorBrewer_1.1-2	rgl_0.108.3	ggplot2_3.3.5	26
## [16] SeuratObject_4.0.4	Seurat_4.0.5		27
##			28
## loaded via a namespace (and not attached):			29
## [1] utf8_1.2.2	reticulate_1.22	tidyselect_1.1.1	30
## [4] RSQLite_2.2.9	htmlwidgets_1.5.4	grid_4.0.3	31
## [7] Rtsne_0.15	pROC_1.18.0	msnse_0.5.0	32
## [10] codetools_0.2-18	ica_1.0-2	future_1.23.0	33
## [13] miniUI_0.1.1.1	withr_2.4.3	colorspace_2.0-2	34
## [16] highr_0.9	knitr_1.36	rstudioapi_0.13	35
## [19] ROCR_1.0-11	tensor_1.5	Rttf2pt1_1.3.9	36
## [22] listenv_0.8.0	labeling_0.4.2	polyclip_1.10-0	37
## [25] bit64_4.0.5	farver_2.1.0	parallelly_1.29.0	38
## [28] vctrs_0.3.8	generics_0.1.1	ipred_0.9-12	39
## [31] xfun_0.28	randomForest_4.6-14	R6_2.5.1	40
## [34] bitops_1.0-7	spatstat.utils_2.2-0	cachem_1.0.6	41
## [37] assertthat_0.2.1	promises_1.2.0.1	scales_1.1.1	42
## [40] nnet_7.3-14	gtable_0.3.0	globals_0.14.0	43
## [43] processx_3.5.2	goftest_1.2-3	timeDate_3043.102	44
## [46] rlang_0.4.12	splines_4.0.3	extrafontdb_1.0	45
## [49] lazyeval_0.2.2	ModelMetrics_1.2.2.2	checkmate_2.0.0	46
## [52] spatstat.geom_2.3-0	yaml_2.2.1	reshape2_1.4.4	47
## [55] abind_1.4-5	backports_1.4.0	httpuv_1.6.3	48
## [58] Hmisc_4.6-0	caret_6.0-90	DiagrammeR_1.0.6.1	49
## [61] extrafont_0.17	tools_4.0.3	lava_1.6.10	50
## [64] ellipsis_0.3.2	spatstat.core_2.3-2	proxy_0.4-26	51
## [67] ggridges_0.5.3	Rcpp_1.0.7	plyr_1.8.6	52
## [70] base64enc_0.1-3	visNetwork_2.1.0	purrr_0.3.4	53
## [73] ps_1.6.0	rpart_4.1-15	deldir_1.0-6	54
## [76] pbapply_1.5-0	zoo_1.8-9	nichenetr_1.0.0	55
## [79] ggrepel_0.9.1	cluster_2.1.0	webshot2_0.0.0.9000	56
## [82] magrittr_2.0.1	data.table_1.14.2	RSpectra_0.16-0	57
## [85] scattermore_0.7	lmtest_0.9-39	RANN_2.6.1	58
## [88] fitdistrplus_1.1-6	matrixStats_0.61.0	hms_1.1.1	59
## [91] patchwork_1.1.1	mime_0.12	evaluate_0.14	60
## [94] xtable_1.8-4	jpeg_0.1-9	gridExtra_2.3	61
## [97] compiler_4.0.3	tibble_3.1.6	KernSmooth_2.23-20	62
## [100] crayon_1.4.2	websocket_1.4.1	htmltools_0.5.2	63

## [103] mgcv_1.8-33	later_1.3.0	tzdb_0.2.0	64
## [106] Formula_1.2-4	tidyR_1.1.4	lubridate_1.8.0	65
## [109] DBI_1.1.1	MASS_7.3-53	Matrix_1.3-4	66
## [112] readr_2.1.1	cli_3.1.0	gower_0.2.2	67
## [115] igraph_1.2.9	pkgconfig_2.0.3	foreign_0.8-81	68
## [118] plotly_4.10.0	spatstat.sparse_2.0-0	recipes_0.1.17	69
## [121] foreach_1.5.1	chromote_0.0.0.9003	prodlim_2019.11.13	70
## [124] stringr_1.4.0	digest_0.6.29	sctransform_0.3.2	71
## [127] RcppAnnoy_0.0.19	spatstat.data_2.1-0	rmarkdown_2.11	72
## [130] leiden_0.3.9	htmlTable_2.3.0	uwot_0.1.11	73
## [133] curl_4.3.2	shiny_1.7.1	lifecycle_1.0.1	74
## [136] nlme_3.1-153	jsonlite_1.7.2	viridisLite_0.4.0	75
## [139] limma_3.46.0	fansi_0.5.0	pillar_1.6.4	76
## [142] lattice_0.20-41	fastmap_1.1.0	httr_1.4.2	77
## [145] survival_3.2-7	glue_1.5.1	fdrttool_1.2.17	78
## [148] png_0.1-7	iterators_1.0.13	bit_4.0.4	79
## [151] class_7.3-17	stringi_1.7.6	blob_1.2.2	80
## [154] caTools_1.18.2	latticeExtra_0.6-29	memoise_2.0.1	81
## [157] irlba_2.3.5	e1071_1.7-9	future.apply_1.8.1	82