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

13 - Compare with Mafb-KO IM

2022-01-31 00:16:35 +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

1 Description

In this analysis, we compared Mafb-KO sample to Control sample. The analysis peppine was the same as previous ones. We found new Mafb-KO specific population (population "Enriched in Mafb-KO"). Using signature scoring, we showed the "Enriched in Mafb-KO" population had low IM signature score than Control IM population but had higher cMo signature, indicating they were in a immature differeniation state.

The IM and cMo signature scoring was made with VISION package (DeTomaso et al., 2019). Briefly, The IM- and cMo-specific gene signatures were calculated with previously published scRNAseq data (Schyns et al., 2019) by comparing either IM or cMo population to all other cell types in the dataset using FindMarker function (Seurat). The genes with logFC > 1 and only positively regulated ones were considered as IM or cMo signature. The signatures were then used to calculate the scores for each cells in the IM and Mafb KO-enriched clusters with VISION package. The scores were stored in Seurat object and plotted with Seurat package.

GSEA results were obtained using GSEA desktop (Subramanian et al., 2005) with the input data described in the following codes. The results could be found in the same analysis folder.

2 Load packages and data

```
suppressMessages({
  library(Seurat)
  library(ggplot2)
  library(multcomp)
})

results <- readRDS(file = "../12-cMAF_and_Mafb_deficient_IM/All_samples_
   Maf.seuratObject.Rds")

# we will work on control and Mafb-KO samples, so subset:
so <- subset(results, subset = group == c("HT5-Control", "HT7-MAFb-KO"))
```

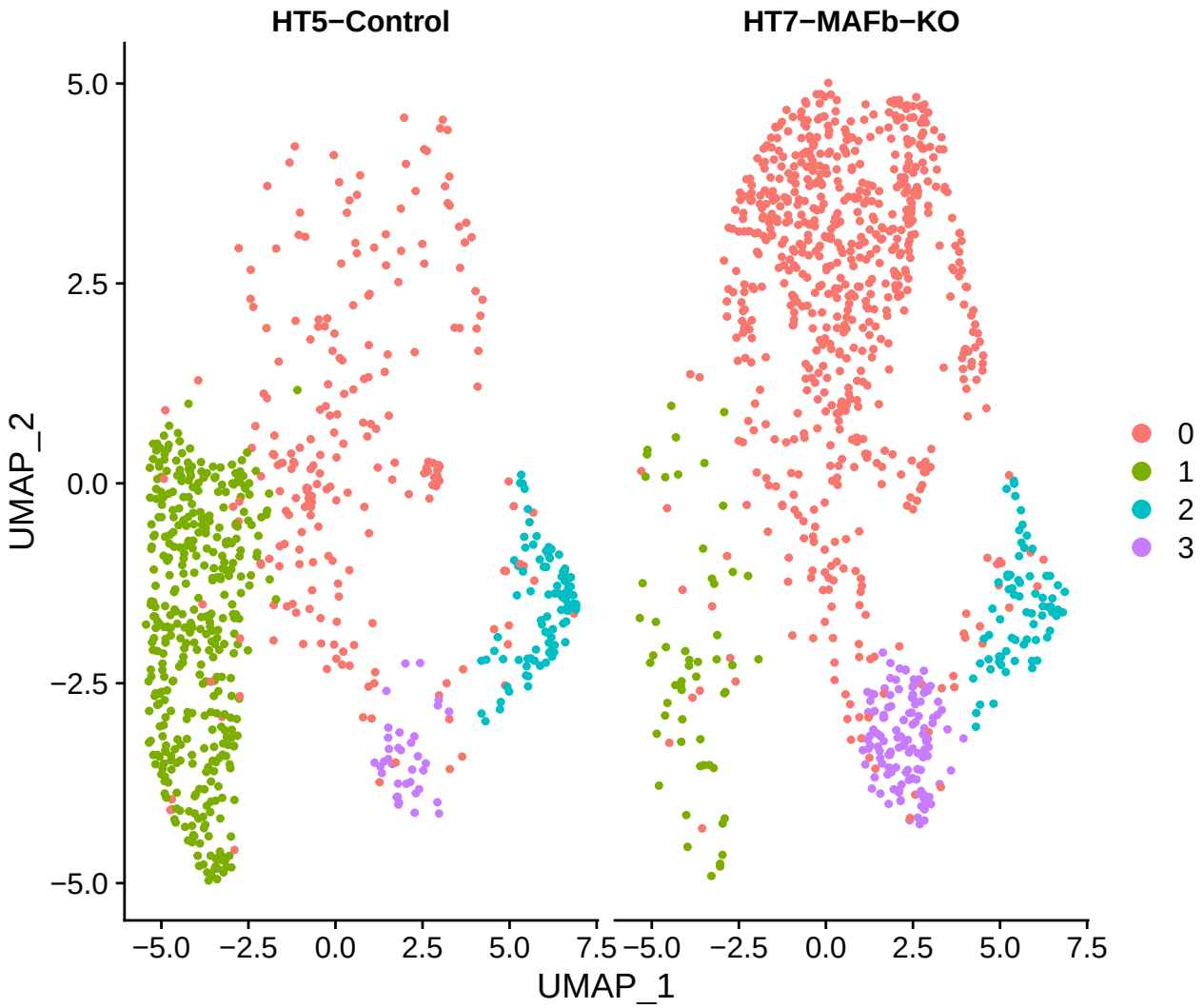
3 Re process data with only Control and Mafb-KO samples

3.1 Re-process data

```
so <- NormalizeData(so, verbose=FALSE)
so <- FindVariableFeatures(so, selection.method = "vst", nfeatures = 1000,
    verbose=FALSE) # we focus less variable genes.
so <- ScaleData(so, features = rownames(so), verbose=FALSE)
so <- RunPCA(so, features = VariableFeatures(so), verbose=FALSE)
so <- RunTSNE(so, dims = 1:8, verbose=FALSE)
so <- RunUMAP(so, dims = 1:8, verbose=FALSE)
```

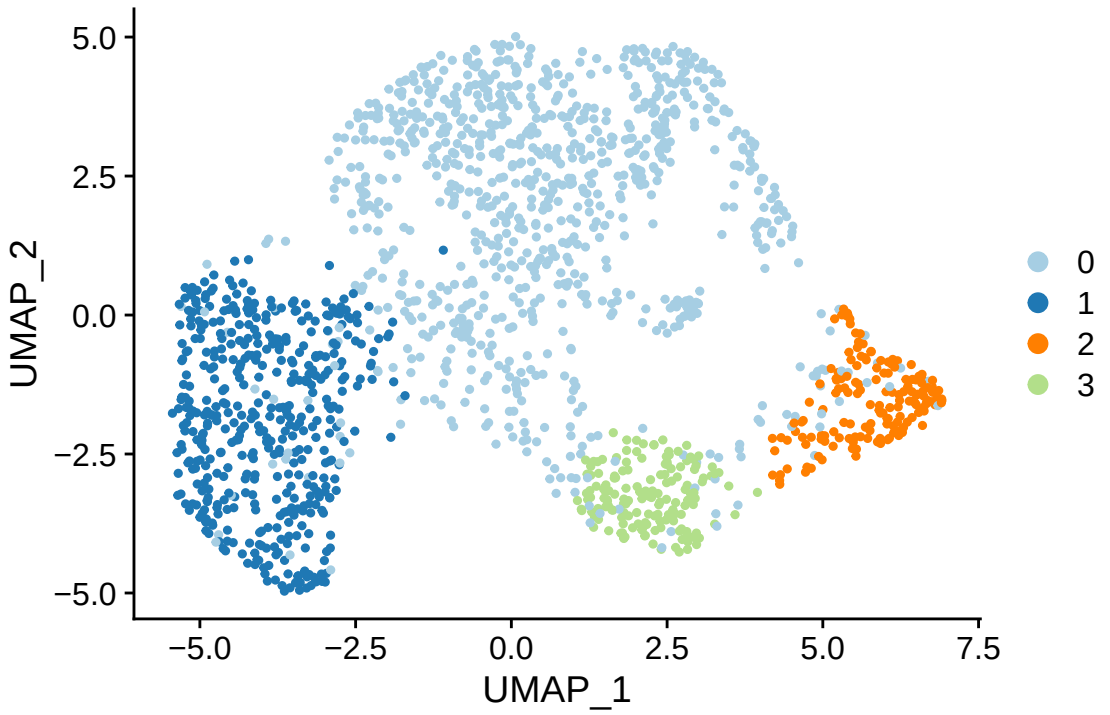
```
so <- FindNeighbors(so, dims = 1:8, verbose = FALSE)
so <- FindClusters(so, resolution = 0.15, verbose = FALSE)
```

```
DimPlot(so, split.by = "group")
```

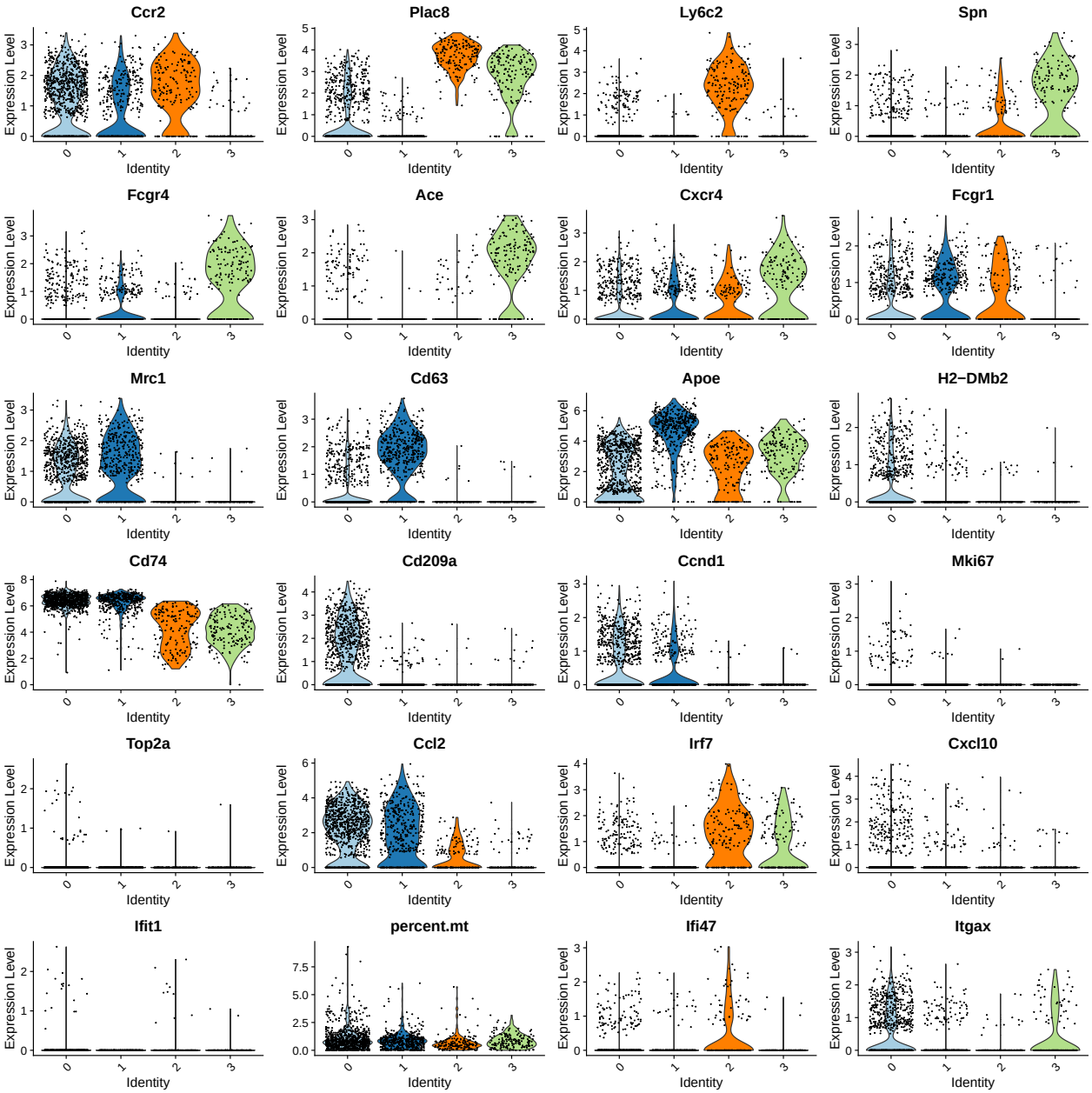


4 Compare population composition

```
pal3 <- c(
  "#A6CEE3", # cMo
  "#1F78B4", # pMo
  "#FF7F00", # Intermediate
  "#B2DF8A", # MHCII IM
  "#33A02C", # CD206 IM
  "#CAB2D6", # Mafb- neo
  "#ededed"# Unknown
)
DimPlot(so, cols = pal3[1:4])
```




```
VlnPlot(
  so,
  features = c("Ccr2", "Plac8" ,"Ly6c2", "Spn",
                               "Fcgr4", "Ace", "Cxcr4",
                               "Fcgr1", "Mrc1", "Cd63", "Apoe",
                               "H2-DMb2", "Cd74", "Cd209a",
                               "Ccnd1","Mki67", "Top2a", "Ccl2",
                               "Irf7", "Cxcl10", "Ifit1", "percent.mt", "
                                  Ifi47", "Itgax") ,
        ncol = 4, cols = pal3[1:4] )
```

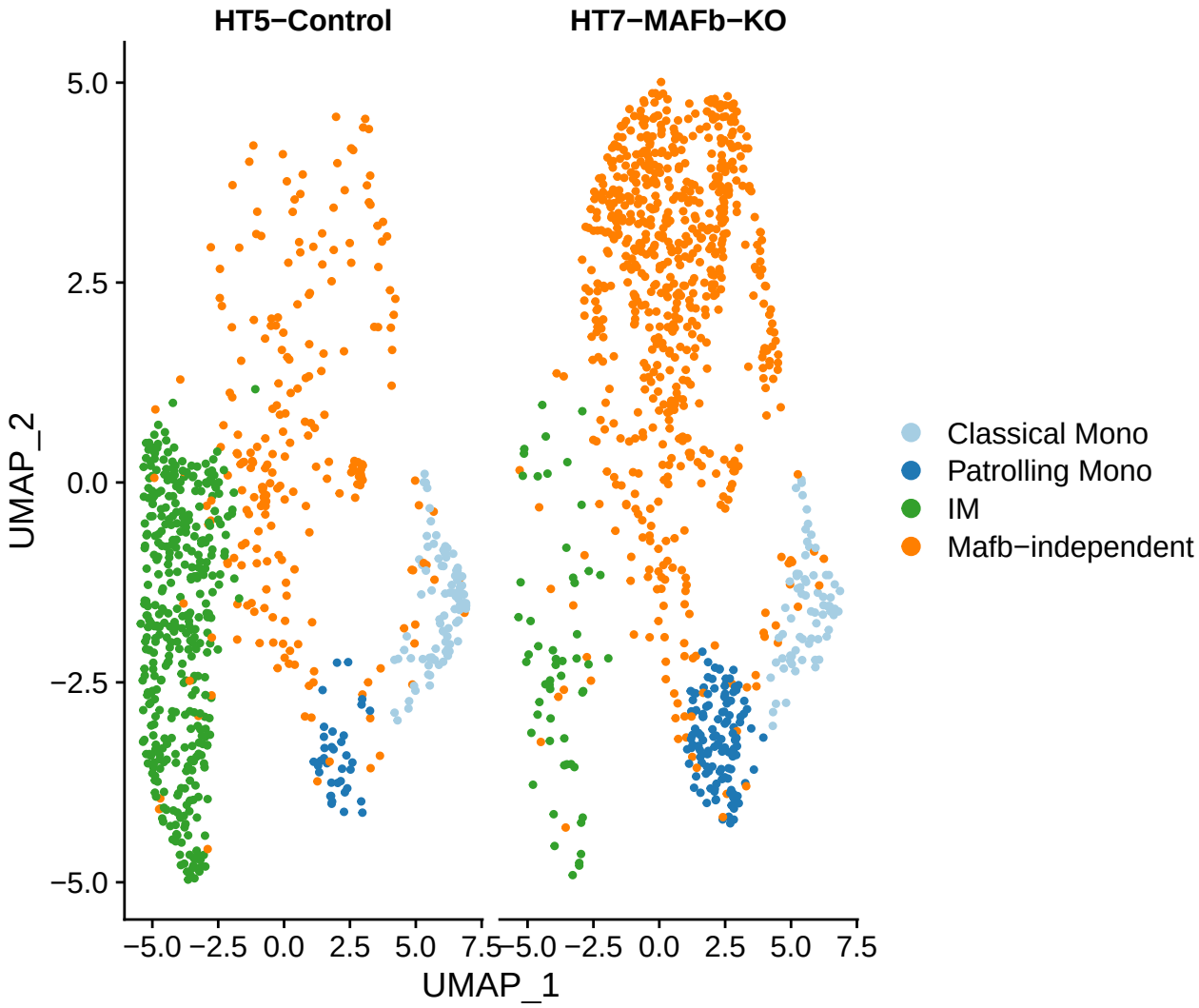


cluster 0: Mafb-independent Cluster 1: IM Cluster 2: Classical Monocytes Cluster 3: Patrolling Monocytes

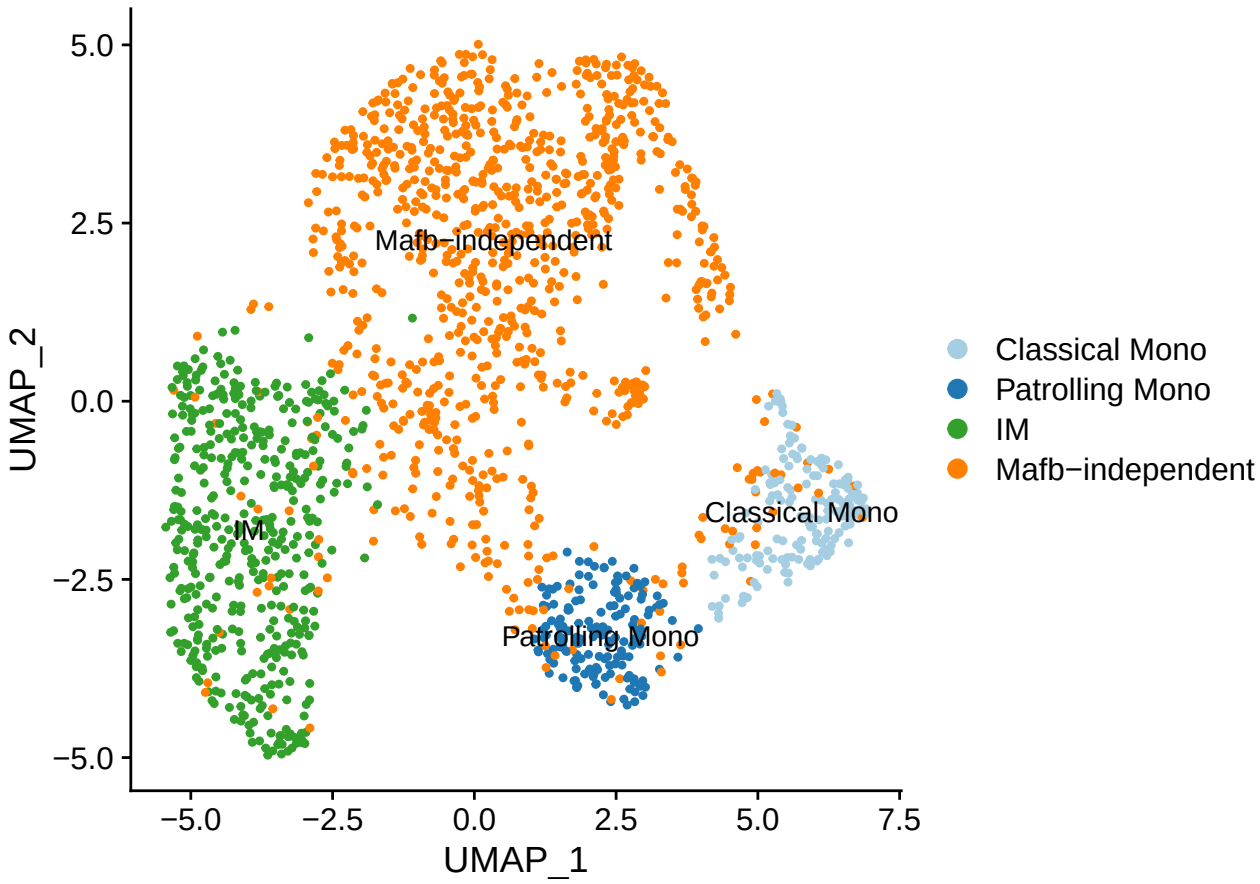
```
so$cell.type2 <- factor(Idents(so), labels = c("Mafb-independent", "IM", "
    Classical␣Mono", "Patrolling␣Mono"))
so$cell.type2 <- factor(so$cell.type2, levels = c("Classical␣Mono", "
    Patrolling␣Mono", "IM", "Mafb-independent"))
Idents(so) <- "cell.type2"
```

```
pal4 <- c(
  "#A6CEE3", # cMo
  "#1F78B4", # pMo
  "#33A02C", # CD206 IM
   "#FF7F00" # Mafb- neo
)
DimPlot(so, cols = pal4, split.by = "group"
```

```
)
```




```
DimPlot(so, label = T, cols = pal4)
```

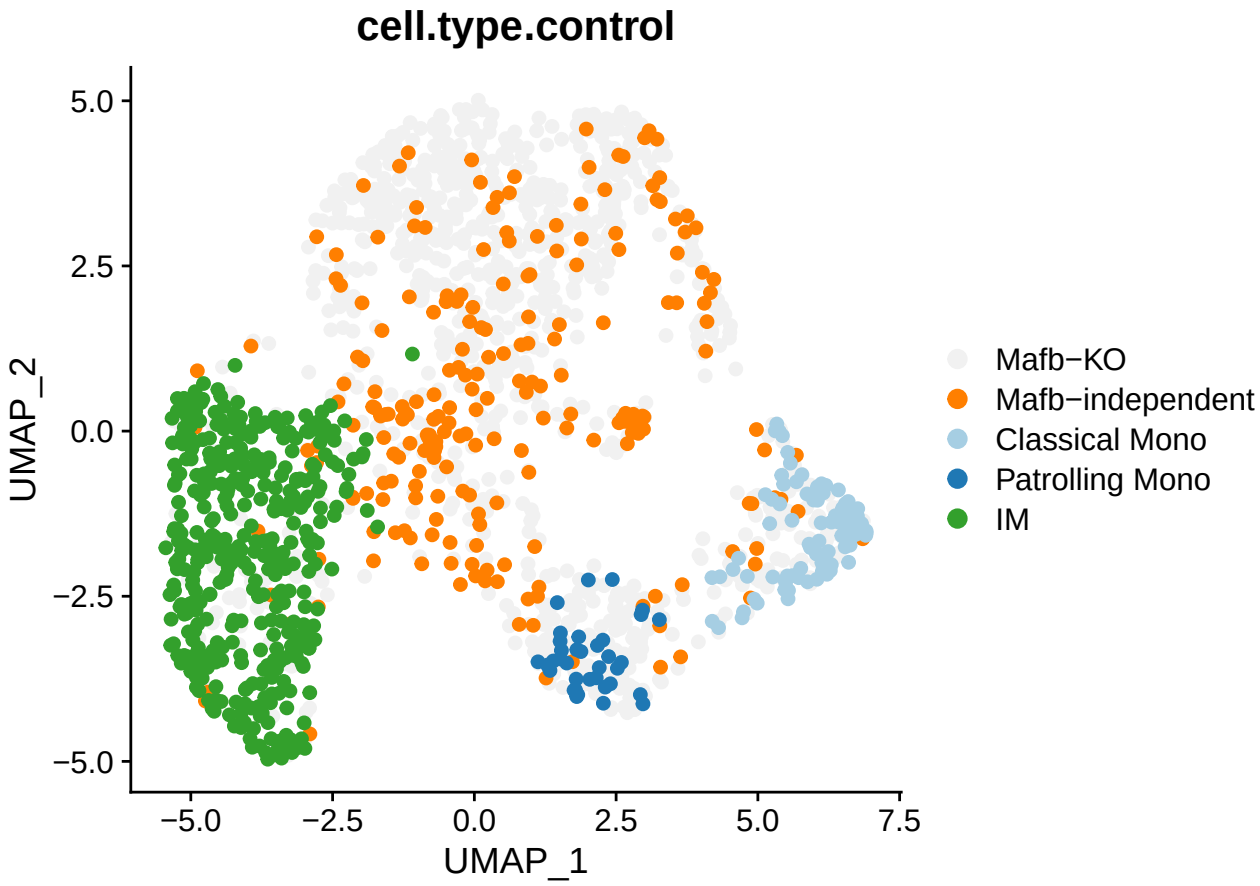



```
ggsave(filename = "../Figures/UMAPplot_Ctl_AND_MafbKO_with_legend.pdf",
       width = 7, height =5)
```

Plot cell in colors but only for one of two samples

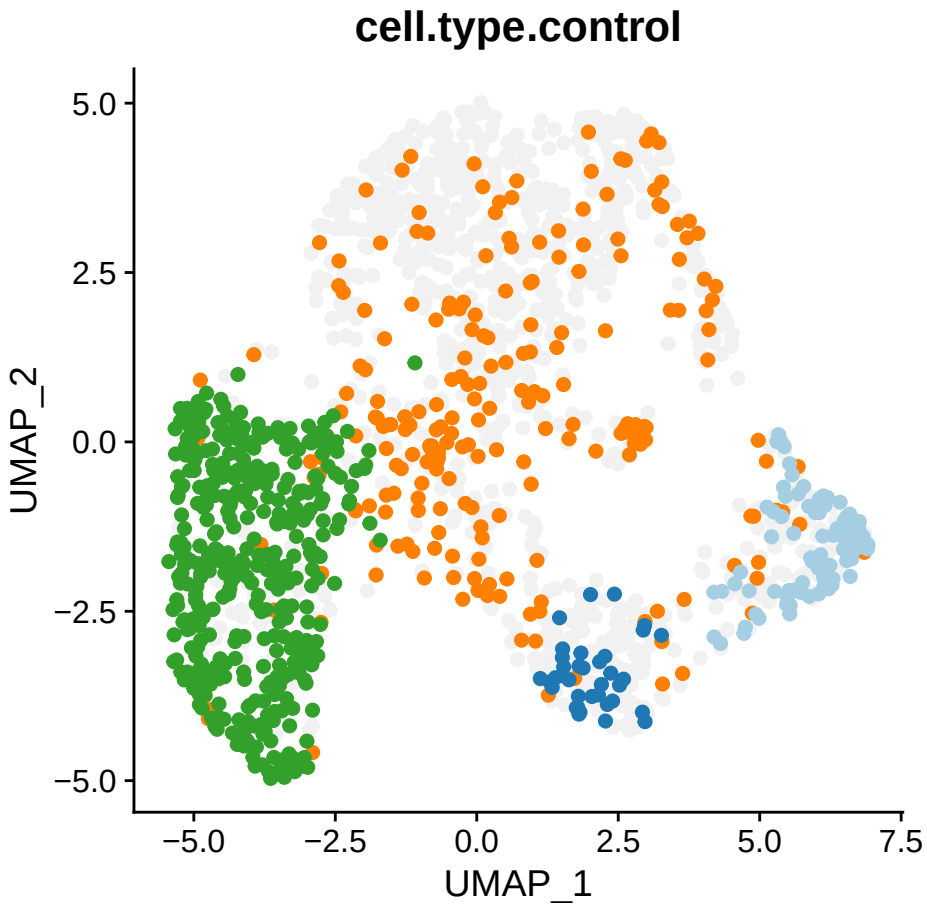
```
so$cell.type.control <- as.character(so$cell.type2)
so$cell.type.control[WhichCells(so, expression = group == "HT7-MAFb-KO")]
   <- "Mafb-KO"
so$cell.type.control <- factor(so$cell.type.control, levels = c("Classical
   ␣Mono", "Patrolling␣Mono", "IM", "Mafb-independent", "Mafb-KO"))
```

```
pal4.control <- c(
  "#F1F1F1", # grey for another sample
  "#FF7F00", # Mafb- neo
  "#A6CEE3", # cMo
  "#1F78B4", # pMo
  "#33A02C" # CD206 IM
)
DimPlot(so, cols = pal4.control, group.by = "cell.type.control", pt.size =
    2,
order = c("IM", "Patrolling␣Mono", "Classical␣Mono", "Mafb-independent",
   "Mafb-KO")
)
```




```
ggsave(filename = "../Figures/UMAPplot_Ctl_IM_with_legend.pdf",
       width = 7, height =5)
```

```
DimPlot(so, cols = pal4.control, group.by = "cell.type.control", pt.size =
    2,
order = c("IM", "Patrolling Mono", "Classical Mono", "Mafb-independent",
   "Mafb-KO")
) + NoLegend()
```

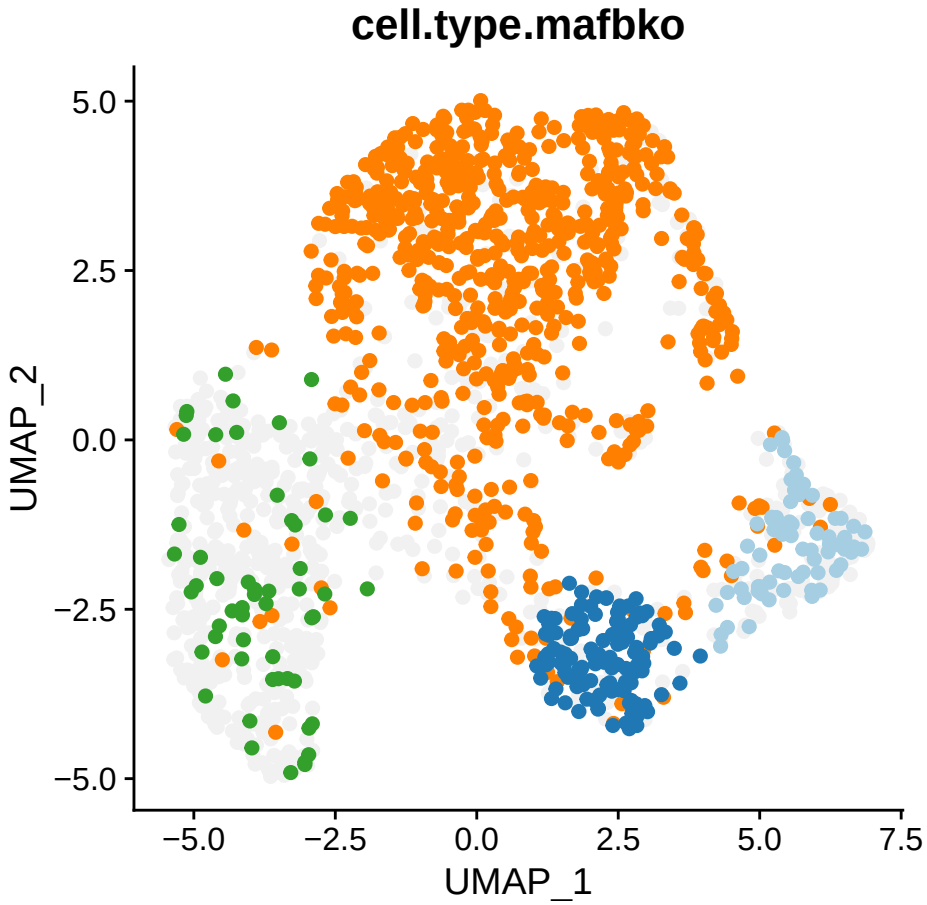



```
ggsave(filename = "../Figures/UMAPplot_Ctl_(vsMafbKO)_no_legend.pdf",
       width = 5, height = 5)
```

Plot Mafb-KO:

```
so$cell.type.mafbko <- as.character(so$cell.type2)
so$cell.type.mafbko[WhichCells(so, expression = group == "HT5-Control")]
    <- "Control"
so$cell.type.mafbko <- factor(so$cell.type.mafbko, levels = c("Classical␣
    Mono", "Patrolling␣Mono", "IM", "Mafb-independent", "Control"))
```

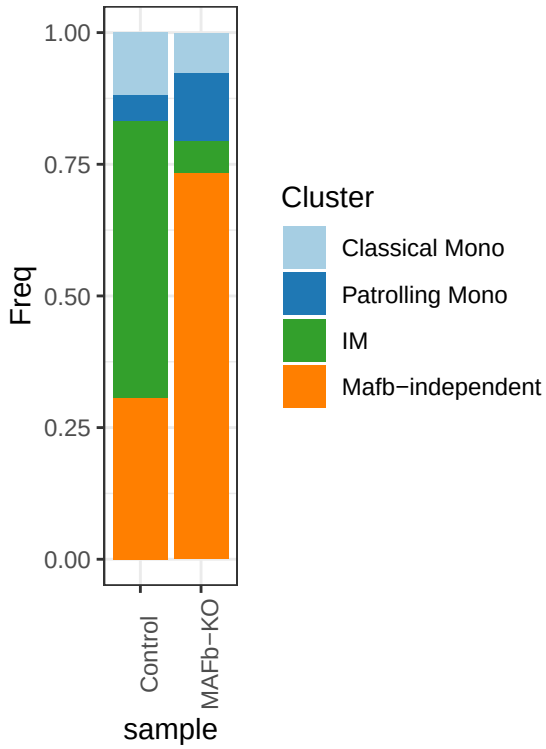
```
pal4.control <- c(
  "#F1F1F1", # grey for another sample
  "#FF7F00", # Mafb- neo
  "#A6CEE3", # cMo
  "#1F78B4", # pMo
  "#33A02C" # CD206 IM
)
DimPlot(so, cols = pal4.control, group.by = "cell.type.mafbko", pt.size =
    2,
order = c("IM", "Patrolling␣Mono", "Classical␣Mono", "Mafb-independent",
    "Mafb-KO")
) + NoLegend()
```




```
ggsave(filename = "../Figures/UMAPplot_MafbKO_(vsControl)_no_legend.pdf",
       width = 5, height = 5)
```

See population frequencies:

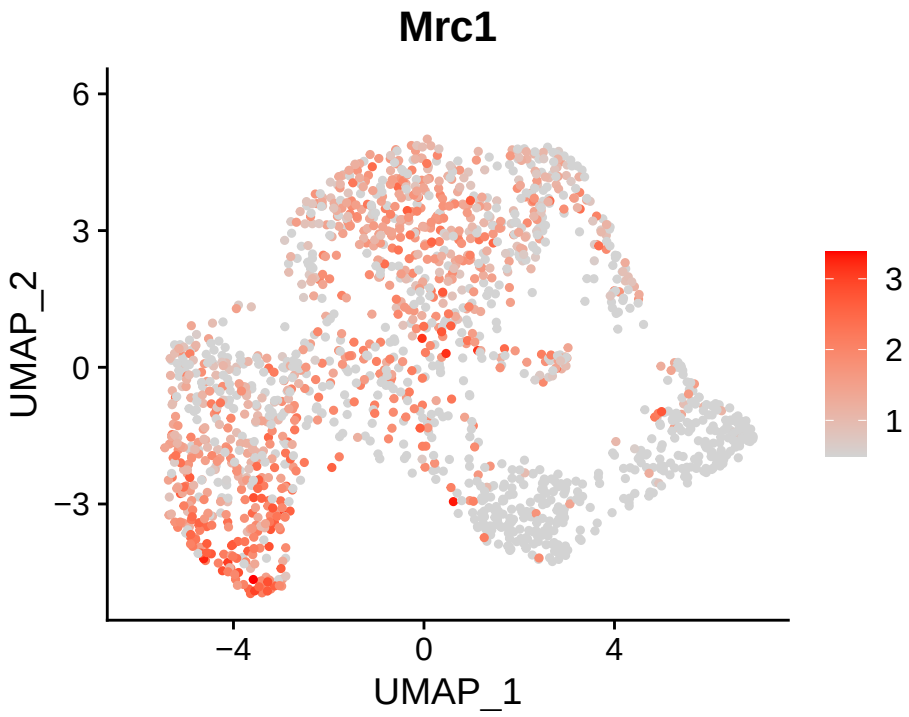
```
source("../R/SeuratFreqTable.R")
freq.celltype.list <- list(
  `Control` = Seurat2CellFreqTable(subset(so, subset = group == "HT5-
     Control"), slotName = "cell.type2"),
  `MAFb-KO` = Seurat2CellFreqTable(subset(so, subset = group == "HT7-MAFb-
     KO"), slotName = "cell.type2")
)

source("../R/barChart.R")
barChart(freq.celltype.list) + labs(fill = "Cluster") + scale_fill_manual(
   values = pal4) + theme(axis.text.x = element_text(angle = 90))
```




```
ggsave(filename = "../Figures/Barplot_Ctl_MafbKO_population_frequency.pdf"
    ,
        width = 3, height = 4)
```

Show CD206+ IM markers

```
FeaturePlot(so, features = "Mrc1", cols = c("lightgray", "red"), min.
    cutoff = 0.5)
```

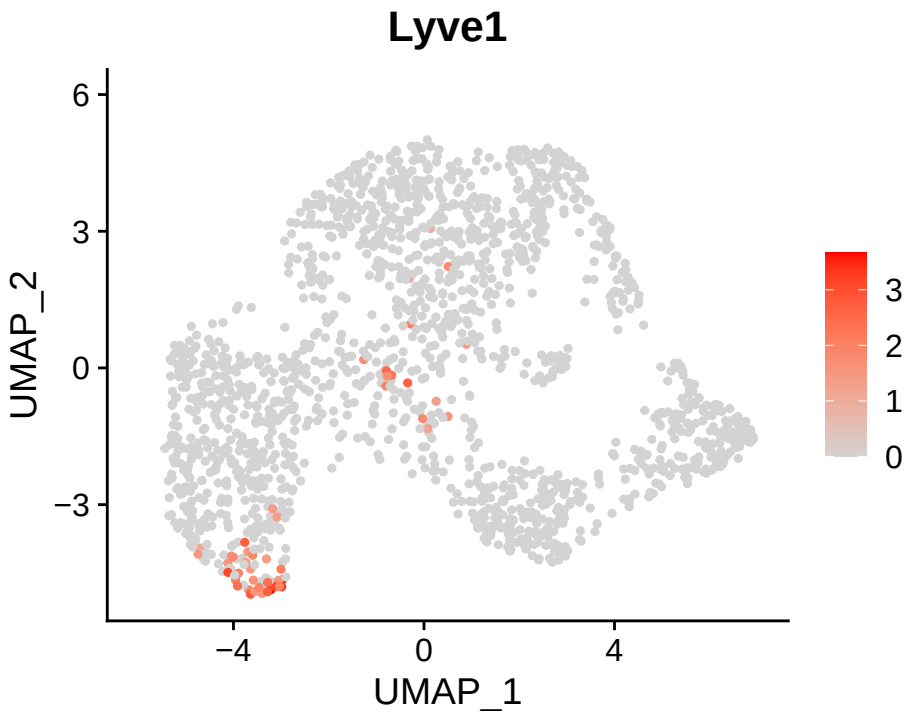



```
ggsave(filename = "../Figures/VlnPlot_Mrc1_in_all_samples.pdf", width = 5,
    height = 4)
```

1

```
FeaturePlot(so, features = "Lyve1", cols = c("lightgray", "red"))
```

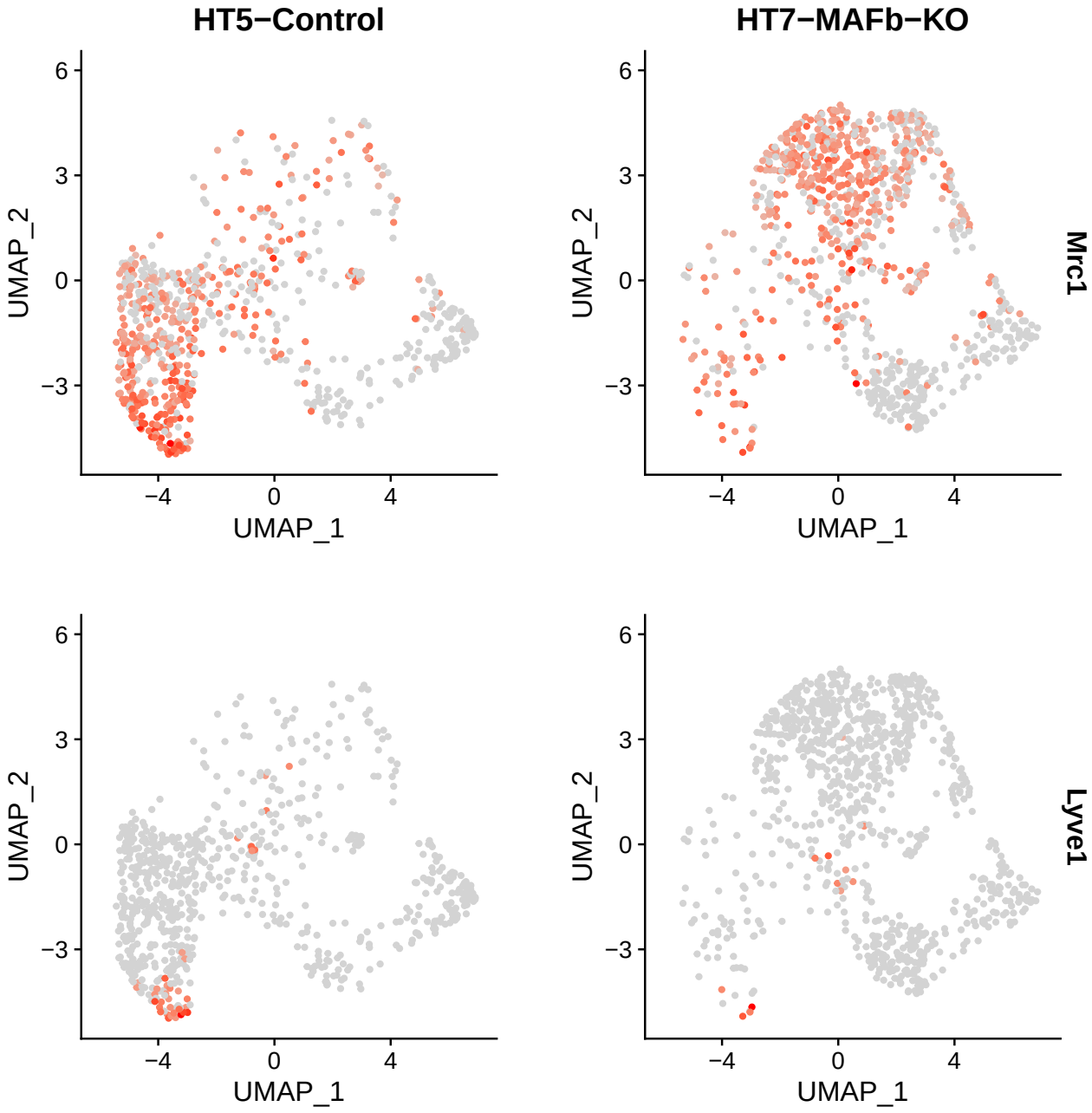
1




```
ggsave(filename = "../Figures/VlnPlot_Lyve1_in_all_samples.pdf", width =
   5, height = 4)
```

1

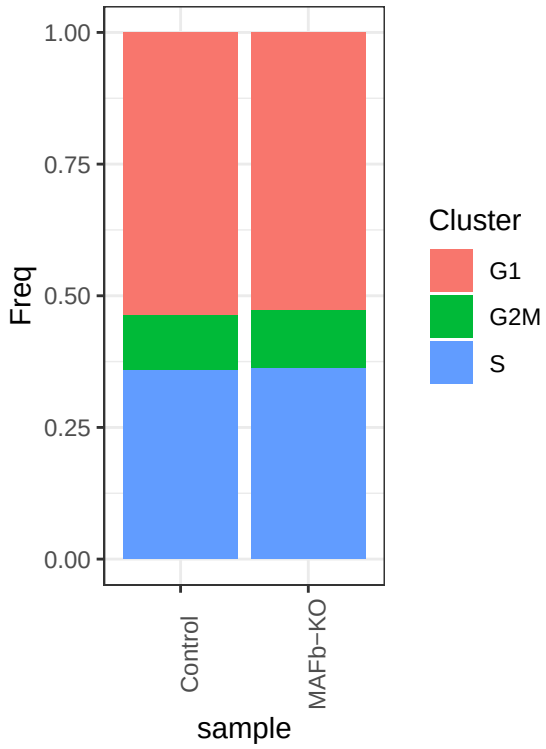
```
FeaturePlot(so, features = c("Mrc1", "Lyve1"), cols = c("lightgray", "red"
    ), split.by = "group")
```



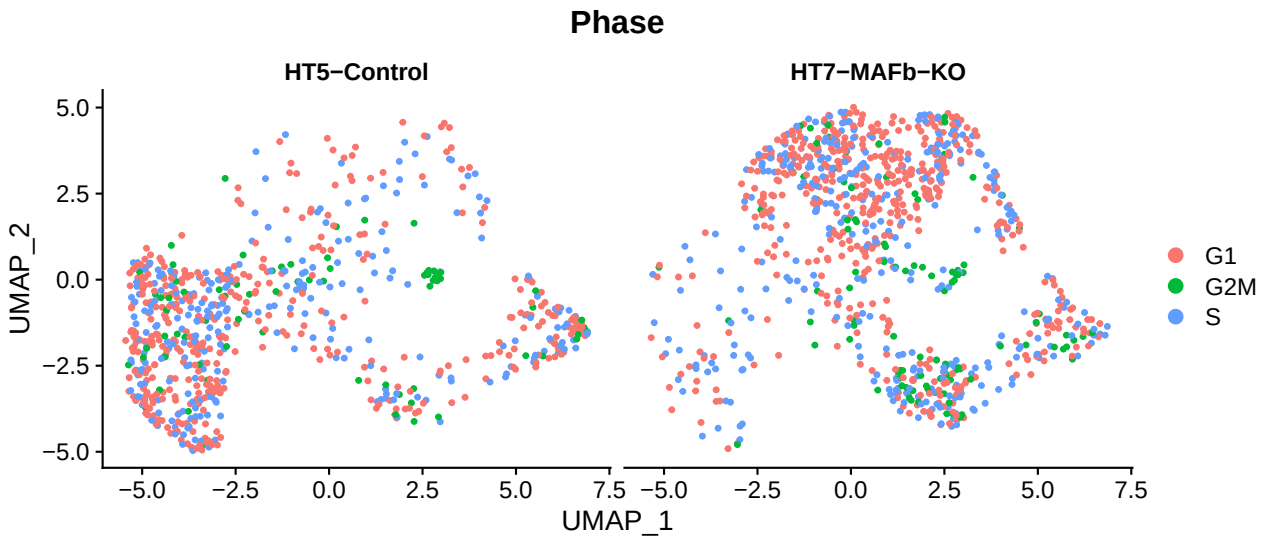
5 Proliferation comparison

```
freq.celltype.list <- list(
  `Control` = Seurat2CellFreqTable(subset(so, subset = group == "HT5-
     Control"), slotName = "Phase"),
  `MAFb-KO` = Seurat2CellFreqTable(subset(so, subset = group == "HT7-MAFb-
     KO"), slotName = "Phase")
)
```

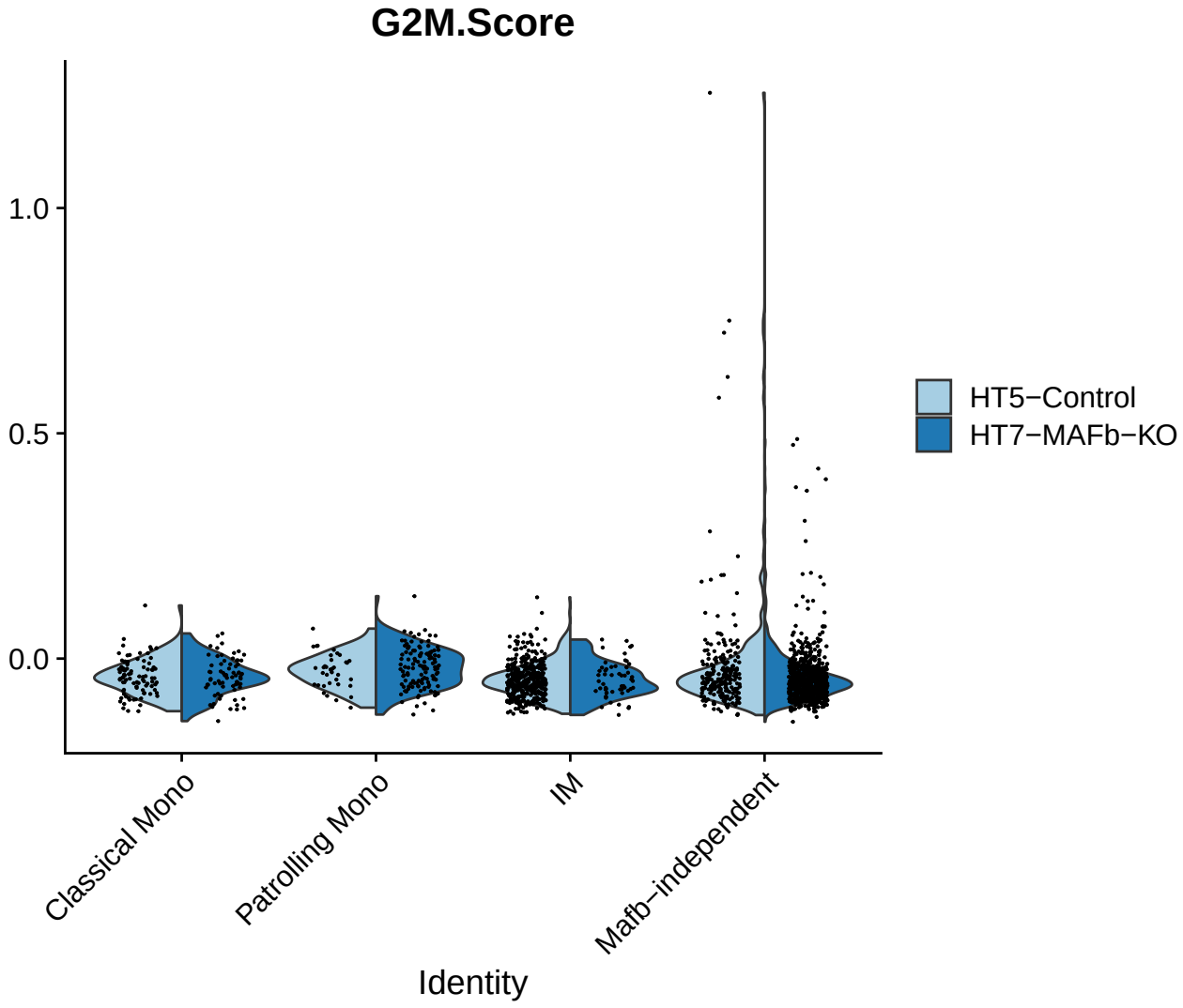
```
barChart(freq.celltype.list) + labs(fill = "Cluster") +  theme(axis.text.x
    = element_text(angle = 90))
```




```
DimPlot(so, group.by = "Phase", split.by = "group")
```




```
VlnPlot(so, features = "G2M.Score", split.by = "group", cols = pal3, split
    .plot = TRUE)
```



6 Different population between Mafb-KO and Control sample

Let's focus on the Mafb-deficient population in Mafb-deficient sample.

```
neo_IM <- subset(so, subset = cell.type2 %in% c("Mafb-independent", "IM"))
```

6.1 DE genes between Mafb- neo and IM population

```
library(dplyr)
Neo_vs_IM <- FindMarkers(so,
                         ident.1 = "Mafb-independent",
                         ident.2 = c("IM"),
                         logfc.threshold = 0,
                         verbose = FALSE)


# keep only adj p value < 0.05 and logFC > 0.5 as significant markers.
Neo_vs_IM.markers <- Neo_vs_IM[Neo_vs_IM$p_val_adj < 0.05 & abs(Neo_vs_IM$
    avg_log2FC) > 0.5, ]
```

```
Neo_vs_IM.markers <- Neo_vs_IM.markers[order(Neo_vs_IM.markers$avg_log2FC,
    decreasing = TRUE), ]
nrow(Neo_vs_IM.markers)
```

```
## [1] 216
```

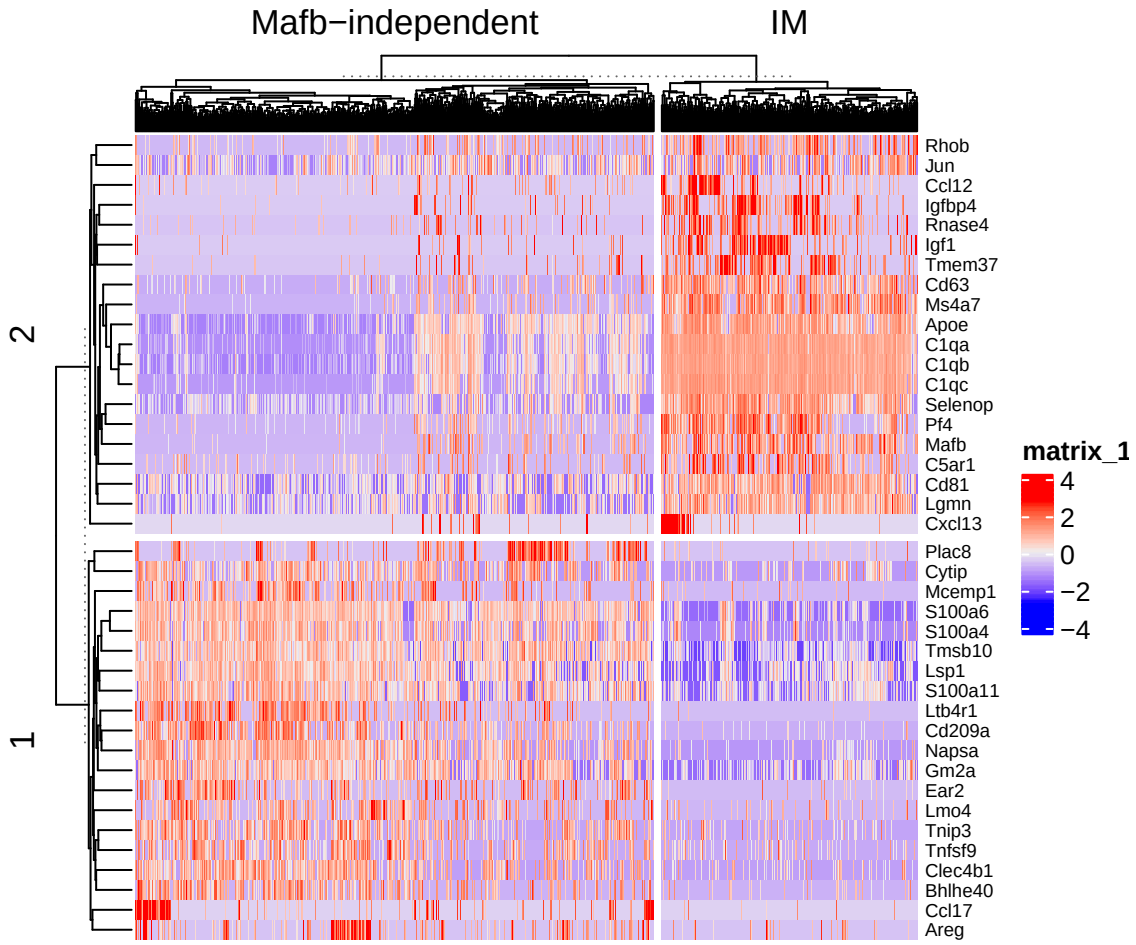
```
write.csv(Neo_vs_IM.markers ,file = "./Mafb-deficient_vs_IM.DEgenes.
   results.csv", quote = FALSE)
```

Let's show the top 20 of each side:

```
Neo_vs_IM.markers.top20 <- Neo_vs_IM.markers[c(1:20, (nrow(Neo_vs_IM.
   markers)-19):(nrow(Neo_vs_IM.markers))), ]
```

```
library(ComplexHeatmap)
mat <- GetAssayData(neo_IM)[rownames(Neo_vs_IM.markers.top20), ]
mat.scale <- t(scale(t(as.matrix(mat))))
```

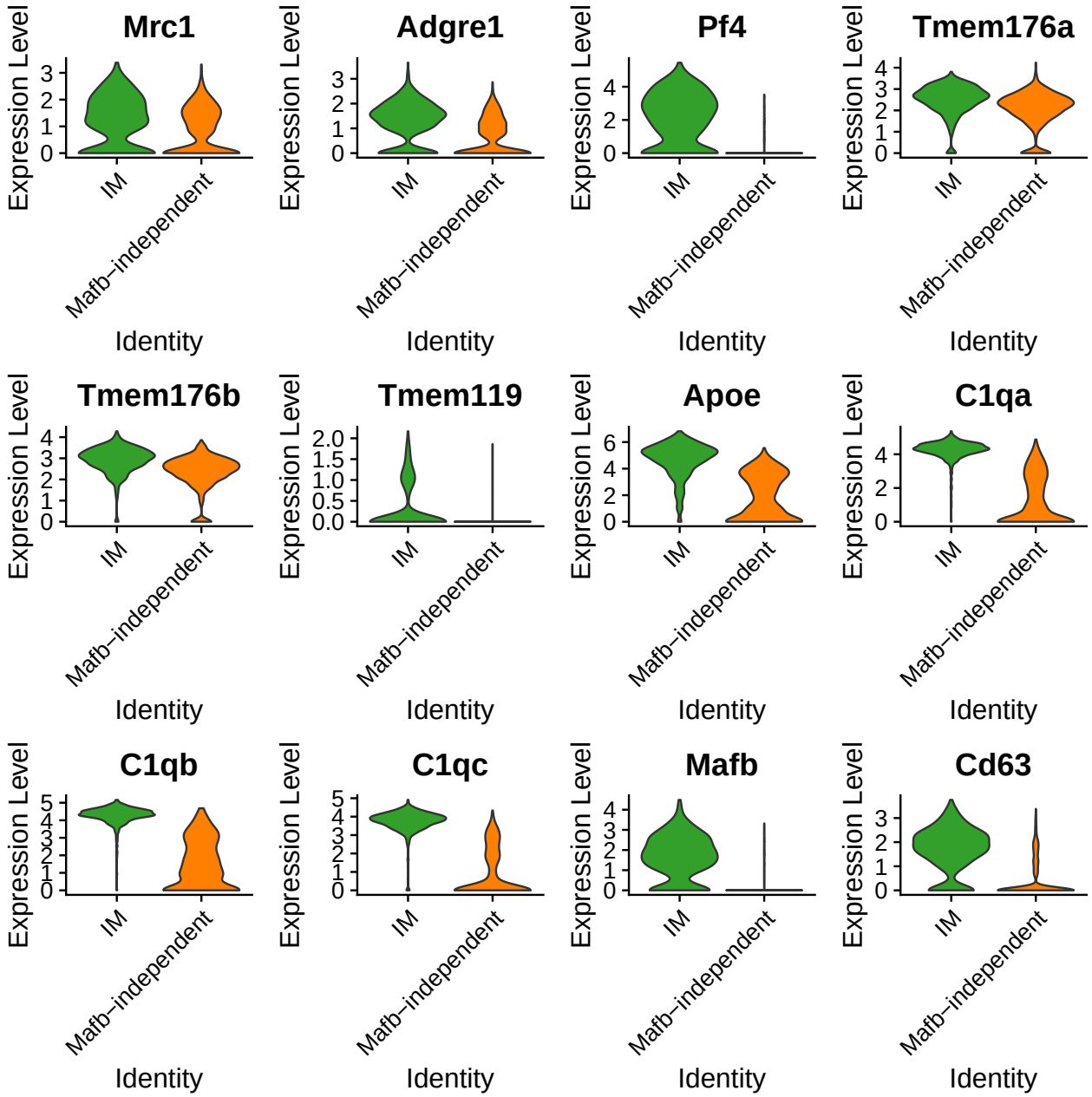
```
hp <- Heatmap(mat.scale, show_row_names = TRUE, show_column_names = FALSE,
        row_names_gp = gpar ( fontsize = 7),
        column_split = factor(neo_IM$cell.type2),
        km = 2)
hp <- draw(hp)
```



```
pdf(file = "../Figures/Heatmap_IM_vs_Mafb-neo.pdf", width = 6, height = 5)
hp
dev.off()
```

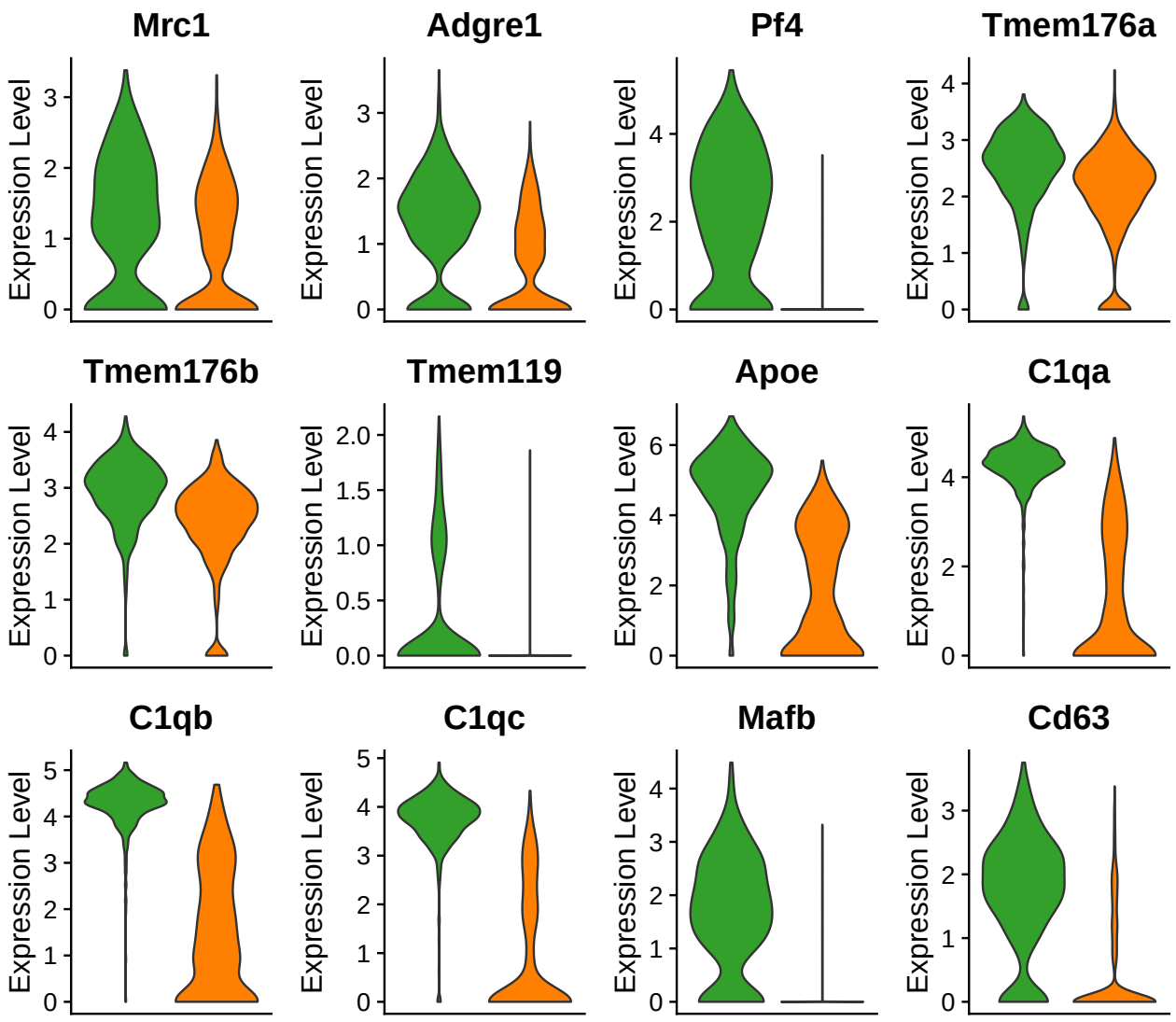
Show in vlnplot

```
p <- VlnPlot(neo_IM, features = c("Mrc1", "Adgre1", "Pf4", "Tmem176a", "
   Tmem176b","Tmem119", "Apoe", "C1qa", "C1qb", "C1qc", "Mafb", "Cd63"),
   group.by = "cell.type2", cols = c("#33A02C", "#FF7F00"), ncol = 4, pt.
   size = 0)
p
```



Show vlnplot without label:

```
p & theme(axis.title.x=element_blank(),
          axis.text.x=element_blank(),
          axis.ticks.x=element_blank())
```




```
ggsave(filename = "../Figures/Vlnplot_IMID_genes_in_Ctrl_MafbKO.pdf",
       width = 8, height = 8)
```

6.2 GO enrichment analysis with DE genes

```
suppressMessages(library(clusterProfiler))
source("../R/entrez2symbol.R")
source("../R/replaceEntrezID.R")
```

6.2.1 GO enrichment analysis of up-regulated DE genes in Mafb-deficient population

```
DE.MafbKO <- Neo_vs_IM.markers[Neo_vs_IM.markers$avg_log2FC > 0, ]
symb <- rownames(DE.MafbKO)
```

```
de_entrez <- bitr( geneID = symb, fromType = "SYMBOL", toType = "ENTREZID"
    , OrgDb = "org.Mm.eg.db", drop = TRUE ) $ ENTREZID
result.enrichGO <- enrichGO(de_entrez, OrgDb = "org.Mm.eg.db", ont = "BP")
result.enrichGO <- replaceEntrezID(result.enrichGO, organism = "mmu")
# write.csv(result.enrichGO@result, file = "./Results_enrichment/enrichGO_
    DE_Mafb_KO_vsIM.csv")
result.enrichGO@result
```

```
## # A tibble: 2,119 x 9
##    ID     Description   GeneRatio BgRatio  pvalue p.adjust  qvalue
    geneID  Count
##    <chr>  <chr>         <chr>     <chr>     <dbl>    <dbl>   <dbl> <chr
    >   <int>
##  1 GO:00~ positive reg~ 14/107    435/23~ 1.25e-8  2.66e-5 2.03e-5
    Tnfsf9~    14
##  2 GO:00~ integrin-med~ 7/107     95/233~ 2.78e-7  2.94e-4 2.25e-4
    Itgb7/~     7
##  3 GO:00~ regulation o~ 12/107    427/23~ 6.27e-7  4.43e-4 3.38e-4
    Tnfsf9~    12
##  4 GO:00~ leukocyte mi~ 11/107    360/23~ 8.39e-7  4.44e-4 3.39e-4
    Ccl17/~    11
##  5 GO:00~ positive reg~ 12/107    449/23~ 1.06e-6  4.50e-4 3.44e-4
    Tnfsf9~    12
##  6 GO:00~ cytokine sec~ 6/107     76/233~ 1.36e-6  4.81e-4 3.67e-4 Srgn
    /C~     6
##  7 GO:00~ negative reg~ 6/107     84/233~ 2.46e-6  7.44e-4 5.68e-4
    Plaur/~     6
##  8 GO:00~ positive reg~ 9/107     265/23~ 3.74e-6  9.19e-4 7.02e-4
    Tnfsf9~     9
##  9 GO:20~ negative reg~ 6/107     91/233~ 3.93e-6  9.19e-4 7.02e-4
    Plaur/~     6
## 10 GO:00~ leukocyte ce~ 10/107    345/23~ 4.34e-6  9.19e-4 7.02e-4
    Tnfsf9~    10
## # ... with 2,109 more rows
```

6.2.2 GO enrichment analysis of up-regulated DE genes in IMs

```
DE.IM <- Neo_vs_IM.markers[Neo_vs_IM.markers$avg_log2FC < 0, ]
symb <- rownames(DE.IM)
de_entrez <- bitr( geneID = symb, fromType = "SYMBOL", toType = "ENTREZID"
    , OrgDb = "org.Mm.eg.db", drop = TRUE ) $ ENTREZID
result.enrichGO <- enrichGO(de_entrez, OrgDb = "org.Mm.eg.db", ont = "BP")
result.enrichGO <- replaceEntrezID(result.enrichGO, organism = "mmu")
# write.csv(result.enrichGO@result, file = "./Results_enrichment/enrichGO_
    DE_IM_vsMafbKO.csv")
result.enrichGO@result
```

```
## # A tibble: 2,460 x 9
##    ID     Description  GeneRatio BgRatio   pvalue p.adjust  qvalue
    geneID  Count
##    <chr>  <chr>        <chr>     <chr>     <dbl>    <dbl>   <dbl> <chr
    >   <int>
```

```
##  1 GO:00~ cell chemot~ 14/104    303/23~ 8.04e-11  1.98e-7 1.36e-7
   Arrb2/~    14
##  2 GO:00~ leukocyte c~ 12/104    219/23~ 2.80e-10  2.30e-7 1.58e-7
   Fcgr3/~    12
##  3 GO:00~ myeloid leu~ 12/104    219/23~ 2.80e-10  2.30e-7 1.58e-7
   Fcgr3/~    12
##  4 GO:00~ leukocyte m~ 14/104    360/23~ 7.66e-10  4.71e-7 3.23e-7
   Fcgr3/~    14
##  5 GO:00~ positive re~ 10/104    156/23~ 2.04e- 9  1.00e-6 6.88e-7
   Cx3cr1~    10
##  6 GO:00~ regulation ~ 9/104     121/23~ 3.65e- 9  1.29e-6 8.83e-7 Gas6
   /C~     9
##  7 GO:00~ regulation ~ 11/104    217/23~ 3.66e- 9  1.29e-6 8.83e-7
   Cx3cr1~    11
##  8 GO:00~ positive re~ 8/104     96/233~ 1.14e- 8  3.50e-6 2.40e-6 Gas6
   /C~     8
##  9 GO:00~ ERK1 and ER~ 12/104    325/23~ 2.40e- 8  6.56e-6 4.50e-6
   Arrb2/~    12
## 10 GO:01~ neuroinflam~ 7/104     70/233~ 2.72e- 8  6.68e-6 4.58e-6 Ctsc
   /C~     7
## # ... with 2,450 more rows
```

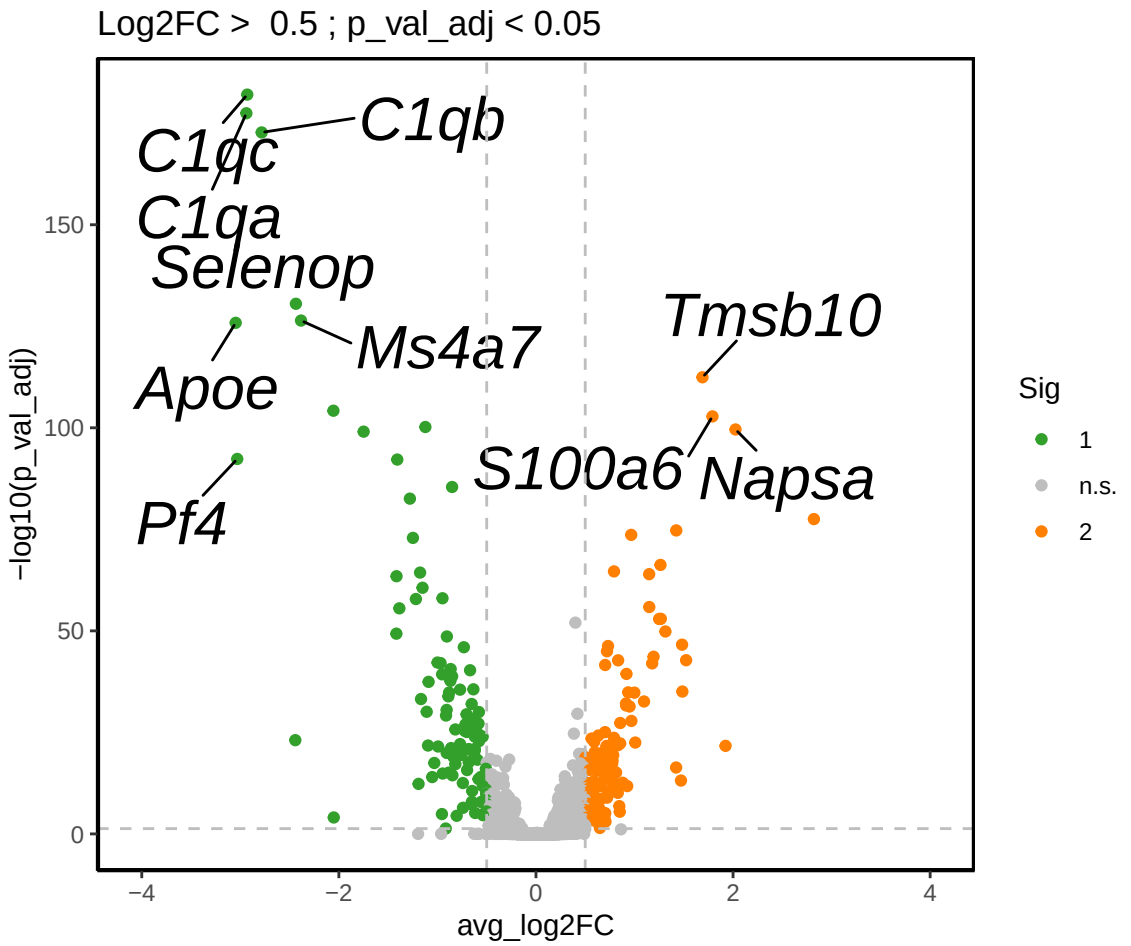
6.2.3 Volcano plot of DE genes

```
suppressMessages({
  library(dplyr)
  library(ggrepel)
  })
```

Let's set a threshold of log2FC and p_val_adj and plot them all:

```
threshold.log2fc <- 0.5
threshold.adjp <- 0.05
Neo_vs_IM.volcano = mutate(Neo_vs_IM,
 Sig=ifelse((abs(Neo_vs_IM$avg_log2FC) > threshold.log2fc)&(Neo_vs_IM$p_
    val_adj < threshold.adjp), "Sig", "n.s."))
# add two colors to 2 sig lists
Neo_vs_IM.volcano$Sig [ Neo_vs_IM.volcano$avg_log2FC < -threshold.log2fc &
    Neo_vs_IM.volcano$p_val_adj < threshold.adjp ] <- "1"

Neo_vs_IM.volcano$Sig [ Neo_vs_IM.volcano$avg_log2FC > threshold.log2fc &
   Neo_vs_IM.volcano$p_val_adj < threshold.adjp ] <- "2"

Neo_vs_IM.volcano$Gene <- rownames(Neo_vs_IM.volcano)
Gene.to.show.ValcanoPlot <- rownames(Neo_vs_IM.volcano[Neo_vs_IM.volcano$
   Sig != "n.s.", ])

p <- ggplot(Neo_vs_IM.volcano, aes(avg_log2FC, -log10(p_val_adj))) + geom_
   point(aes(col=Sig)) + scale_color_manual(values=c( `1`="#33A02C", `n.s
   .`="grey", `2`="#FF7F00"))

# set axis lim:
axis.lim <- max(abs(Neo_vs_IM.volcano$avg_log2FC)) + 1

```

```
p + geom_text_repel(data=filter(Neo_vs_IM.volcano, Gene %in% Gene.to.show.
    ValcanoPlot), size = 8, aes(label=Gene, fontface = "italic"), box.
    padding = 1) + xlim(c(-axis.lim, axis.lim)) + theme_classic() + theme(
    panel.border = element_rect(colour = "black", fill = NA, size = 1)) +
    geom_hline(yintercept = -log10(threshold.adjp), linetype='dashed', col
    = 'grey') + geom_vline(xintercept = c(-threshold.log2fc, threshold.
    log2fc),  linetype='dashed', col = 'grey') + ggtitle(paste("Log2FC > ",
     threshold.log2fc, "; p_val_adj <", threshold.adjp))
```




```
# write.csv(Neo_vs_IM.volcano[Gene.to.show.ValcanoPlot, ] %>% arrange(.,
    desc(avg_log2FC)) ,
#           file = paste("./Mafb-deficient_vs_IM.DEgenes.Log2FC", threshold
    .log2fc, ".adjPval", threshold.adjp, ".results.csv", sep = "") )
```

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

7 GSEA analysis: Mafb-KO population vs IM

```
expr.table <- GetAssayData(neo_IM, slot = "data", assay = "RNA")
expr.table <- as.data.frame(expr.table)
```

```
row.info <- data.frame(NAME=rownames(expr.table), DESCRIPTION=rownames(
    expr.table))
expr.table <- cbind(row.info, expr.table)

# write.table(expr.table, file = "./AssayData_RNA_data_MafbKO_control_IM.
    txt", sep = "\t", quote = FALSE, row.names = FALSE)

# metadata:
cls.table <- matrix(as.character(neo_IM@meta.data$cell.type2), nrow = 1)
# write.table(cls.table, file = "./Class_celltype2_MafbKO_control_IM.cls",
     sep = " ", quote = FALSE, col.names = FALSE, row.names = FALSE) # pay
    attention to the row.names=FALSE, or it will add row 1 to the head.
```

```
# To add to cls file:
cat(paste(length(cls.table), length(unique(as.character(cls.table))), 1),
    "\n",
    "# ", paste(unique(as.character(cls.table)), collapse = " "),
    sep = "")
```

```
# For chip file
all.features <- read.csv(file = "../../../IM-DTR_MAF/counts/scRNAseq/
    Experiment-7-12-21-ScRNA_NGS21-U976/outs/raw_feature_bc_matrix/features
    .tsv.gz", sep = "\t")[, 2]
chip.file <- data.frame(`Probe Set ID`=all.features,
                        `Gene Symbol`=all.features,
                        `Gene Title`=all.features)
# write.table(chip.file, file = "./genename.chip",sep = "\t", quote =
    FALSE, row.names = FALSE )
```

The results can be found in sub directory: GSEA

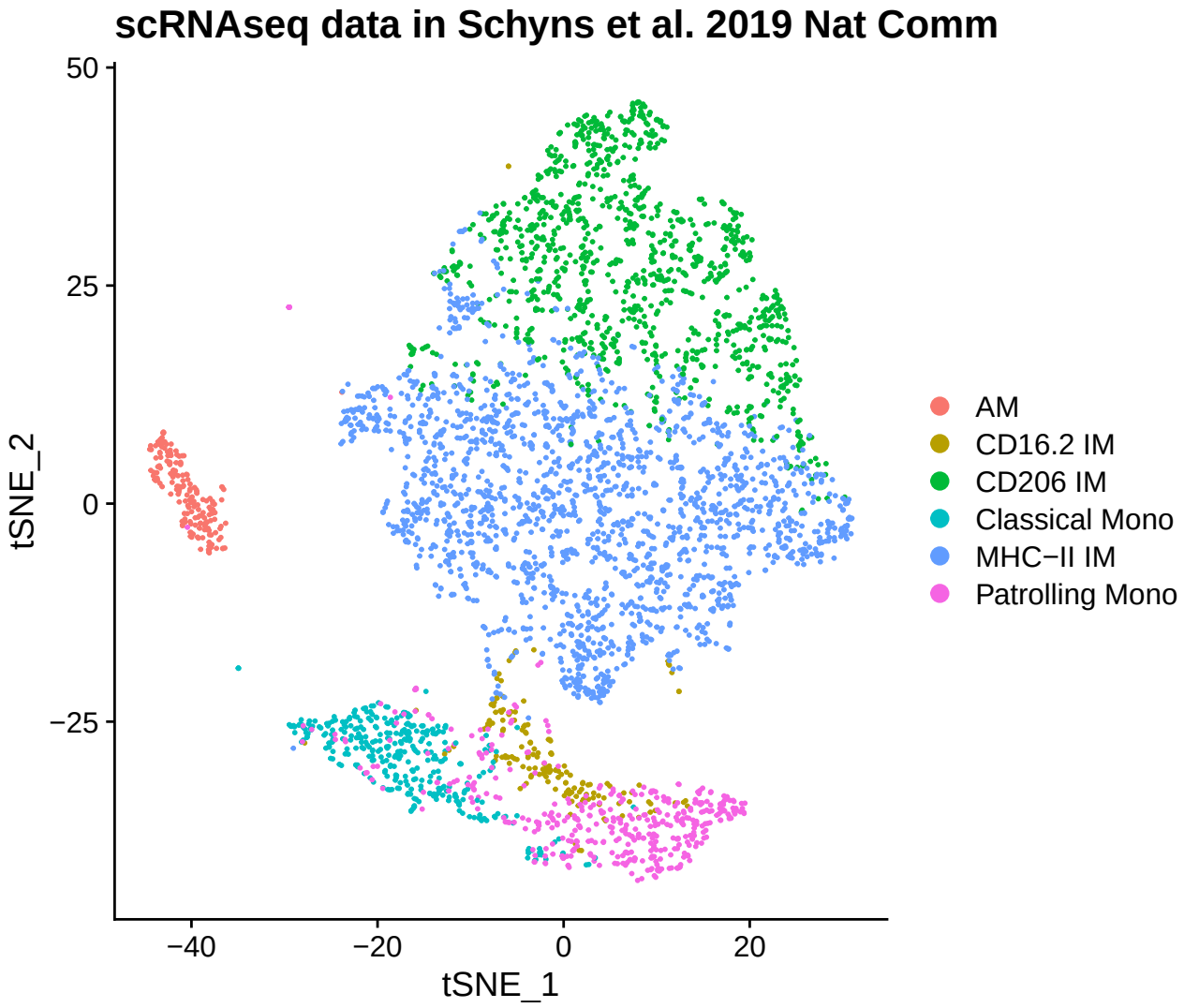
8 Scoring of IM and monocyte signatures in control and Mafb-KO samples

8.1 Load data

We start from our published data (Schyns et al., 2019).

The object "IMss_monocytes_filtered.v3" is the Seurat object used in this report.

```
suppressMessages(library(VISION))
Idents(IMss_monocytes_filtered.v3) <- "population"
DimPlot(IMss_monocytes_filtered.v3) + ggtitle("scRNAseq data in Schyns et
    al. 2019 Nat Comm")
```



8.2 Create signautres for IM, classical monocytes and patrolling monocytes

```
signature.im <- FindMarkers(IMss_monocytes_filtered.v3,
                            ident.1 = c("CD206 IM", "MHC-II IM"),
                            only.pos = TRUE,
                            logfc.threshold = 1, verbose = FALSE)
signature.pmo <- FindMarkers(IMss_monocytes_filtered.v3,
                            ident.1 = "Patrolling Mono",
                            only.pos = TRUE,
                            logfc.threshold = 1, verbose = FALSE)
signature.cmo <- FindMarkers(IMss_monocytes_filtered.v3,
                            ident.1 = "Classical Mono",
                            only.pos = TRUE,
                            logfc.threshold = 1, verbose = FALSE)
```

```
sig.im <- rep(1, length(rownames(signature.im)))
names(sig.im) <- rownames(signature.im)
sig.im <- createGeneSignature(name = "IM", sigData = sig.im)

sig.cmo <- rep(1, length(rownames(signature.cmo)))
```

```
names(sig.cmo) <- rownames(signature.cmo)
sig.cmo <- createGeneSignature(name = "cMo", sigData = sig.cmo)

sig.pmo <- rep(1, length(rownames(signature.pmo)))
names(sig.pmo) <- rownames(signature.pmo)
sig.pmo <- createGeneSignature(name = "pMo", sigData = sig.pmo)

sig.im_mono <- c(sig.im, sig.cmo, sig.pmo)
```

8.3 Signature scoring

```
vis <- Vision(so, signatures = sig.im_mono)

vis <- calcSignatureScores(vis)
head(vis@SigScores)
```

```
##                              IM       cMo       pMo
## AAACCCAAGAGAGAAC-1 0.7238478 2.1288166 2.4711223
## AAACCCAAGCCTATTG-1 1.9573675 1.5396993 1.4073991
## AAACGCTCAATTGCCA-1 2.6581209 0.7510017 0.4236691
## AAAGAACCATTCACCC-1 2.8807122 1.1654191 0.7284988
## AAAGGATTCATCGACA-1 2.1879918 0.8306405 0.4611359
## AAAGGTACAACAGCCC-1 0.5628628 3.6075399 1.9758066
```

8.4 Show signature scores in Seurat object

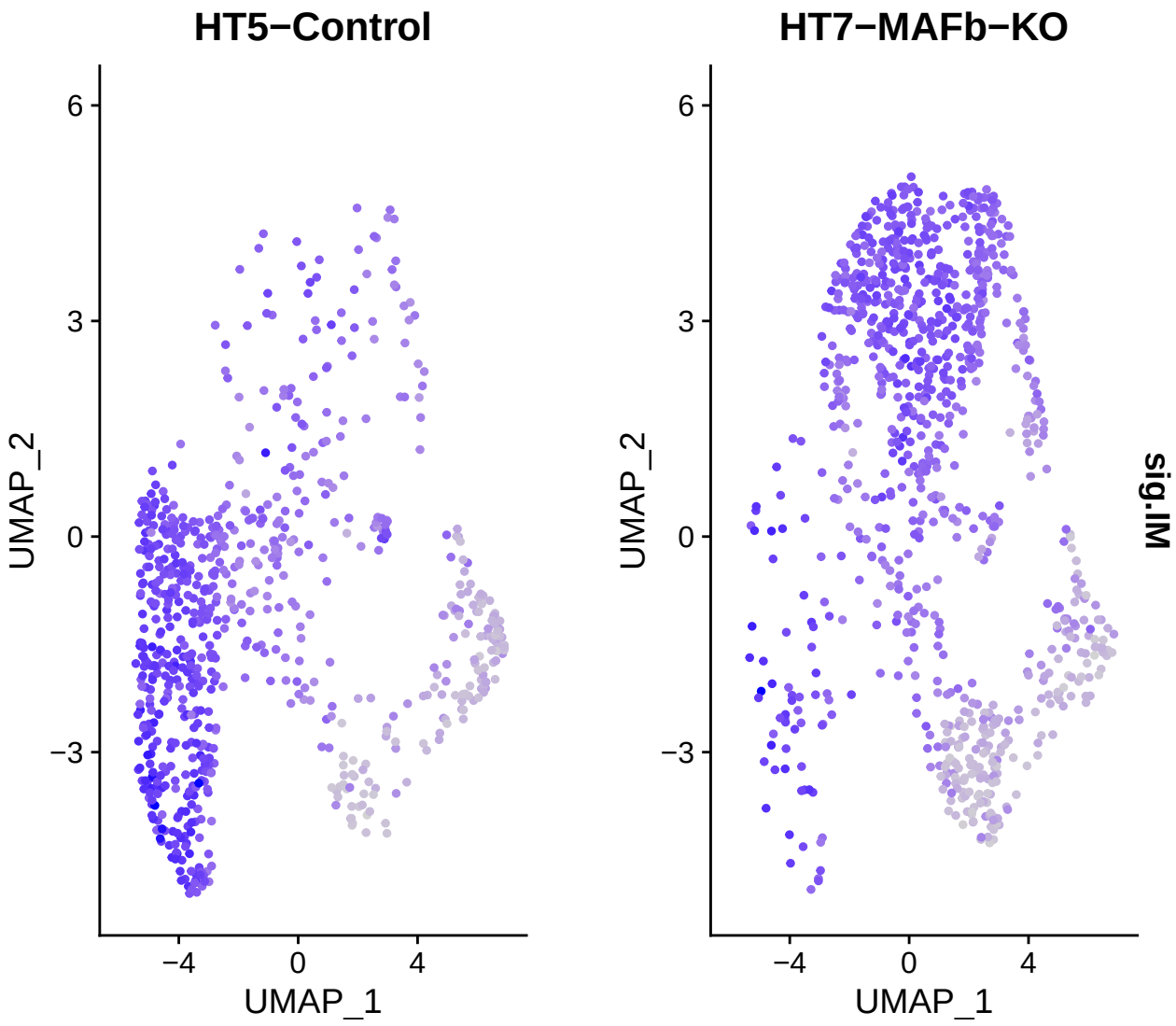
Check if cells are identical

```
identical(colnames(so), rownames(vis@SigScores))
```

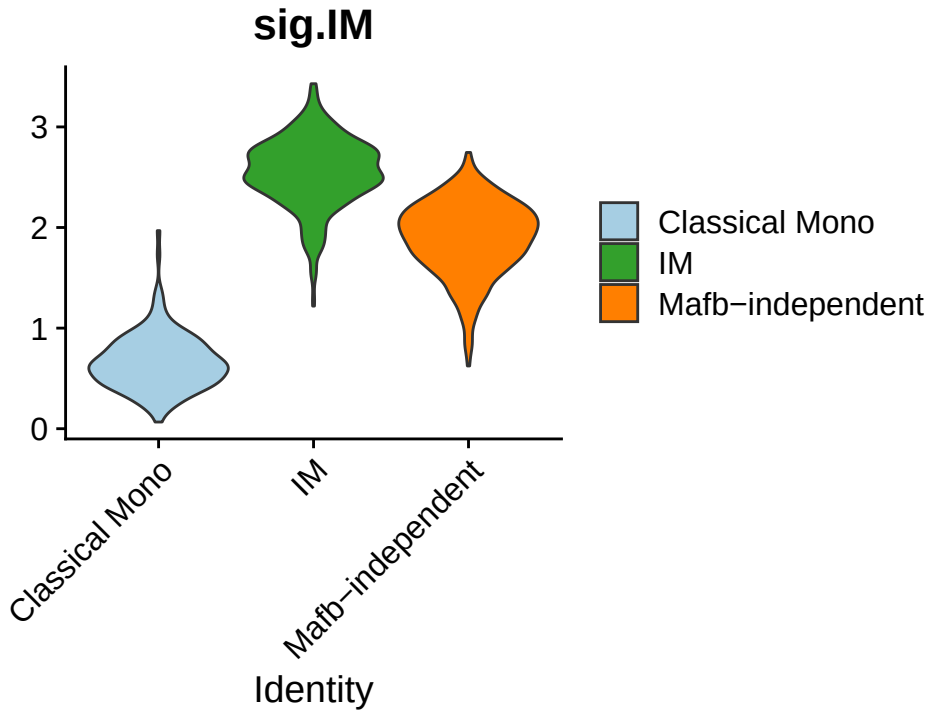
```
## [1] TRUE
```

```
so$sig.IM <- vis@SigScores[, "IM"]
so$sig.pMo <- vis@SigScores[, "pMo"]
so$sig.cMo <- vis@SigScores[, "cMo"]
```

```
FeaturePlot(so, features = "sig.IM", split.by = "group")
```




```
VlnPlot(so, features = "sig.IM", cols = pal4[c(1,3,4)], pt.size = 0,
   idents = c("Classical␣Mono", "IM", "Mafb-independent"))
```

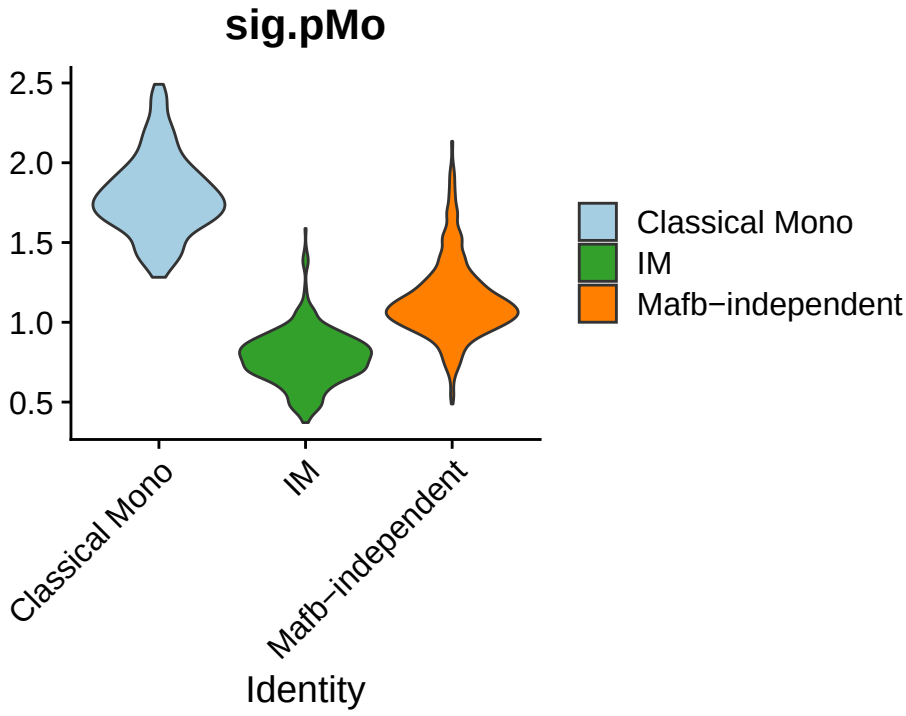



```
so.sub <- subset(so, idents = c("Classical Mono", "IM", "Mafb-independent"
    ))
dt <- data.frame(sig=so.sub$sig.IM, celltype=so.sub$cell.type2)
res.aov <- aov(sig ~ celltype, data = dt)
summary(glht(res.aov, linfct = mcp(celltype = "Tukey")))
```

```
##
##   Simultaneous Tests for General Linear Hypotheses
##
## Multiple Comparisons of Means: Tukey Contrasts
##
##
## Fit: aov(formula = sig ~ celltype, data = dt)
##
## Linear Hypotheses:
##                                        Estimate Std. Error t value Pr
    (>|t|)
## IM - Classical Mono == 0                1.88894    0.03210   58.84   <2
    e-16 ***
## Mafb-independent - Classical Mono == 0  1.23480    0.02990   41.30   <2
    e-16 ***
## Mafb-independent - IM == 0             -0.65414    0.02008  -32.58   <2
    e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
## (Adjusted p values reported -- single-step method)
```

```
ggsave(filename = "../Figures/Vlnplot_sig_IM_in_ctrl_MafbKO.pdf",
       width = 5, height = 4)
```

```
VlnPlot(so, features = "sig.pMo", cols = pal4[c(1,3,4)], pt.size = 0,
   idents = c("Classical Mono", "IM", "Mafb-independent"))
```




```
dt <- data.frame(sig=so$sig.pMo, celltype=so$cell.type2)
res.aov <- aov(sig ~ celltype, data = dt)
summary(glht(res.aov, linfct = mcp(celltype = "Tukey")))
```

```
##
##   Simultaneous Tests for General Linear Hypotheses
##
## Multiple Comparisons of Means: Tukey Contrasts
##
##
## Fit: aov(formula = sig ~ celltype, data = dt)
##
## Linear Hypotheses:
##                                        Estimate Std. Error t value Pr
   (>|t|)
## Patrolling Mono - Classical Mono == 0    0.48802    0.02558   19.08
   <2e-16
## IM - Classical Mono == 0                -1.01759    0.02098  -48.49
   <2e-16
## Mafb-independent - Classical Mono == 0  -0.65801    0.01954  -33.67
   <2e-16
## IM - Patrolling Mono == 0               -1.50561    0.02108  -71.42
   <2e-16
## Mafb-independent - Patrolling Mono == 0 -1.14603    0.01965  -58.33
   <2e-16
## Mafb-independent - IM == 0               0.35958    0.01312   27.40
   <2e-16
```

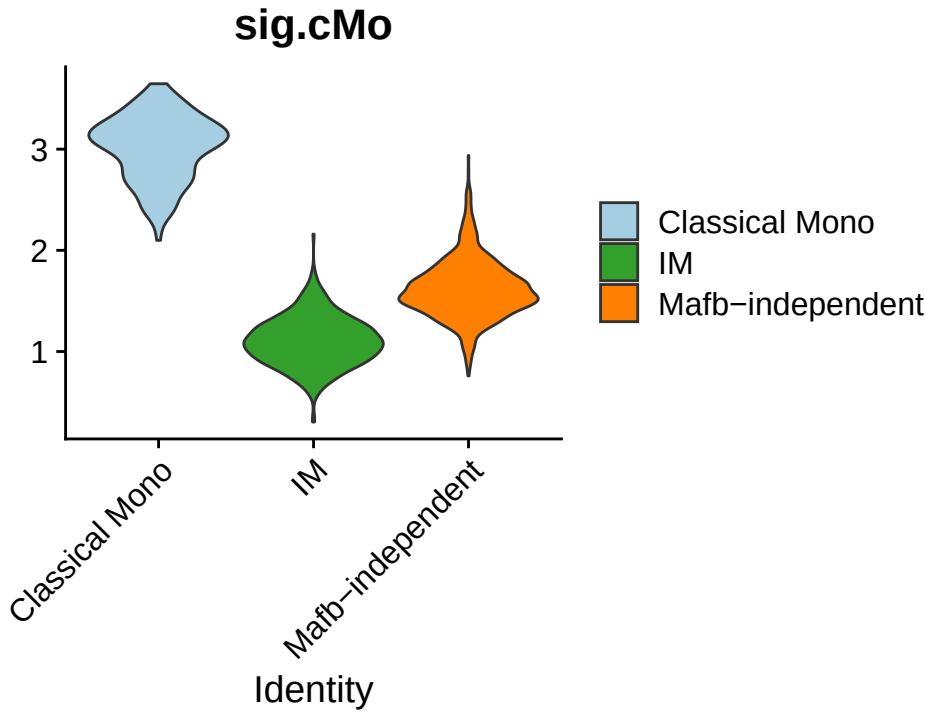
```
##
## Patrolling Mono - Classical Mono == 0   ***
## IM - Classical Mono == 0                ***
## Mafb-independent - Classical Mono == 0  ***
## IM - Patrolling Mono == 0               ***
## Mafb-independent - Patrolling Mono == 0 ***
## Mafb-independent - IM == 0              ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
## (Adjusted p values reported -- single-step method)
```

```
ggsave(filename = "../Figures/Vlnplot_sig_pMo_in_ctrl_MafbKO.pdf",
       width = 5, height = 4)
```

```
dt <- data.frame(sig=so.sub$sig.cMo, celltype=so.sub$cell.type2)
res.aov <- aov(sig ~ celltype, data = dt)
summary(glht(res.aov, linfct = mcp(celltype = "Tukey")))
```

```
##
##   Simultaneous Tests for General Linear Hypotheses
##
## Multiple Comparisons of Means: Tukey Contrasts
##
##
## Fit: aov(formula = sig ~ celltype, data = dt)
##
## Linear Hypotheses:
##                                       Estimate Std. Error t value Pr
   (>|t|)
## IM - Classical Mono == 0              -1.91616    0.02574  -74.43   <2
   e-16 ***
## Mafb-independent - Classical Mono == 0 -1.39959    0.02397  -58.38   <2
   e-16 ***
## Mafb-independent - IM == 0              0.51657    0.01610   32.08   <2
   e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
## (Adjusted p values reported -- single-step method)
```

```
VlnPlot(so, features = "sig.cMo", cols = pal4[c(1,3,4)], pt.size = 0,
   idents = c("Classical␣Mono", "IM", "Mafb-independent"))
```




```
ggsave(filename = "../Figures/Vlnplot_sig_cMo_in_ctrl_MafbKO.pdf",
       width = 5, height = 4)
```

9 Session information

R sesssion:

```
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] parallel  stats4    grid      stats     graphics  grDevices utils
##  [8] datasets  methods   base
##
## other attached packages:
```

```
##  [1] VISION_2.1.0           ggrepel_0.9.1          org.Mm.eg.db_3.12.0
##  [4] AnnotationDbi_1.52.0   IRanges_2.24.1         S4Vectors_0.28.1
##  [7] Biobase_2.50.0         BiocGenerics_0.36.1    clusterProfiler_3
    .18.1
## [10] ComplexHeatmap_2.6.2   dplyr_1.0.7            RColorBrewer_1.1-2
## [13] multcomp_1.4-18        TH.data_1.1-0          MASS_7.3-53
## [16] survival_3.2-7         mvtnorm_1.1-3          ggplot2_3.3.5
## [19] SeuratObject_4.0.4     Seurat_4.0.5
##
## loaded via a namespace (and not attached):
##   [1] utf8_1.2.2            reticulate_1.22        tidyselect_1.1.1
##   [4] RSQLite_2.2.9         htmlwidgets_1.5.4      BiocParallel_1.24.1
##   [7] Rtsne_0.15            scatterpie_0.1.7       munsell_0.5.0
##  [10] codetools_0.2-18      ica_1.0-2              future_1.23.0
##  [13] miniUI_0.1.1.1        withr_2.4.3            fastICA_1.2-3
##  [16] colorspace_2.0-2      GOSemSim_2.16.1        highr_0.9
##  [19] knitr_1.36            rstudioapi_0.13        plumber_1.1.0
##  [22] ROCR_1.0-11           tensor_1.5             pbmcapply_1.5.0
##  [25] DOSE_3.16.0           listenv_0.8.0          labeling_0.4.2
##  [28] polyclip_1.10-0       bit64_4.0.5            farver_2.1.0
##  [31] downloader_0.4        parallelly_1.29.0      vctrs_0.3.8
##  [34] generics_0.1.1        xfun_0.28              R6_2.5.1
##  [37] clue_0.3-60           graphlayouts_0.7.2     rsvd_1.0.5
##  [40] webutils_1.1          spatstat.utils_2.2-0   cachem_1.0.6
##  [43] fgsea_1.16.0          assertthat_0.2.1       promises_1.2.0.1
##  [46] scales_1.1.1          ggraph_2.0.5           enrichplot_1.10.2
##  [49] gtable_0.3.0          Cairo_1.5-12.2         globals_0.14.0
##  [52] goftest_1.2-3         tidygraph_1.2.0        sandwich_3.0-1
##  [55] rlang_0.4.12          GlobalOptions_0.1.2    splines_4.0.3
##  [58] lazyeval_0.2.2        spatstat.geom_2.3-0    BiocManager_1.30.16
##  [61] yaml_2.2.1            reshape2_1.4.4         abind_1.4-5
##  [64] httpuv_1.6.3          qvalue_2.22.0          tools_4.0.3
##  [67] logging_0.10-108      ellipsis_0.3.2         spatstat.core_2.3-2
##  [70] wordspace_0.2-6       ggridges_0.5.3         Rcpp_1.0.7
##  [73] plyr_1.8.6            purrr_0.3.4            rpart_4.1-15
##  [76] deldir_1.0-6          pbapply_1.5-0          GetoptLong_1.0.5
##  [79] viridis_0.6.2         cowplot_1.1.1          zoo_1.8-9
##  [82] swagger_3.33.1        cluster_2.1.0          magrittr_2.0.1
##  [85] data.table_1.14.2     RSpectra_0.16-0        magick_2.7.3
##  [88] scattermore_0.7       DO.db_2.9              circlize_0.4.13
##  [91] lmtest_0.9-39         RANN_2.6.1             fitdistrplus_1.1-6
##  [94] matrixStats_0.61.0    patchwork_1.1.1        mime_0.12
##  [97] evaluate_0.14         xtable_1.8-4           sparsesvd_0.2
## [100] mclust_5.4.8          gridExtra_2.3          shape_1.4.6
## [103] compiler_4.0.3        tibble_3.1.6           KernSmooth_2.23-20
## [106] crayon_1.4.2          shadowtext_0.0.9       htmltools_0.5.2
## [109] ggfun_0.0.4           mgcv_1.8-33            later_1.3.0
## [112] tidyr_1.1.4           DBI_1.1.1              tweenr_1.0.2
## [115] Matrix_1.3-4          permute_0.9-5          cli_3.1.0
## [118] igraph_1.2.9          pkgconfig_2.0.3        rvcheck_0.2.1
## [121] plotly_4.10.0         spatstat.sparse_2.0-0  iotools_0.3-2
## [124] yulab.utils_0.0.4     stringr_1.4.0          digest_0.6.29
## [127] sctransform_0.3.2     RcppAnnoy_0.0.19       vegan_2.5-7
## [130] spatstat.data_2.1-0   rmarkdown_2.11         leiden_0.3.9
```

```
## [133] fastmatch_1.1-3       uwot_0.1.11           loe_1.1                75
## [136] shiny_1.7.1           rjson_0.2.20          lifecycle_1.0.1        76
## [139] nlme_3.1-153          jsonlite_1.7.2        viridisLite_0.4.0      77
## [142] limma_3.46.0          fansi_0.5.0           pillar_1.6.4           78
## [145] lattice_0.20-41       fastmap_1.1.0         httr_1.4.2             79
## [148] GO.db_3.12.1          glue_1.5.1            png_0.1-7              80
## [151] bit_4.0.4             ggforce_0.3.3         stringi_1.7.6          81
## [154] blob_1.2.2            memoise_2.0.1         irlba_2.3.5            82
## [157] future.apply_1.8.1                                                 83
```

References

DeTomaso, D., Jones, M.G., Subramaniam, M., Ashuach, T., Ye, C.J., and Yosef, N. (2019). Functional interpretation of single cell similarity maps. Nature Communications.

Schyns, J., Bai, Q., Ruscitti, C., Radermecker, C., Schepper, S.D., Chakarov, S., Farnir, F., Pirottin, D., Ginhoux, F., Boeckxstaens, G., et al. (2019). Non-classical tissue monocytes and two functionally distinct populations of interstitial macrophages populate the mouse lung. Nature Communications *10*.

Subramanian, A., Tamayo, P., Mootha, V.K., Mukherjee, S., Ebert, B.L., Gillette, M.A., Paulovich, A., Pomeroy, S.L., Golub, T.R., Lander, E.S., et al. (2005). Gene set enrichment analysis: A knowledge-based approach for interpreting genome-wide expression profiles. Proceedings of the National Academy of Sciences of the United States of America.