SCAVENGE-vignette

Fulong Yu

2022-01-19

Overview

This vignette covers the main function and workflow of SCAVENGE. The standard processed input data including fine-mapped variants and single-cell epigenomic profiles. For fine-mapped variants of the trait of interest, we typically need information of genomic locations of variants and their corresponding posterior propability of causality. A peak-by-cell matrix of scATAC-seq profiles is needed. To walk through the workflow of SCAVENGE, we provided a blood cell trait of monocyte count and a 10X PBMC dataset as an example.

Load the required packages

```
devtools::load_all()
library(SCAVENGE)
library(chromVAR)
library(gchromVAR)
library(BuenColors)
library(SummarizedExperiment)
library(data.table)
library(BiocParallel)
library(BSgenome.Hsapiens.UCSC.hg19)
library(dplyr)
set.seed(9527)
```

Load example data

The PBMC data was processed using ArchR package. The peak-by-cell count matrix and corresponding meta data were extracted and stored in a RangedSummarizedExperiment object (for more details please follow our paper).

```
trait_file <- paste0(system.file('inst/extdata', package='SCAVENGE'), "/mono.PP001.bed")
pbmc5krda <- paste0(system.file('inst/rda', package='SCAVENGE'), "/pbmc5k_SE.rda")
load(pbmc5krda)
```

gchromVAR analysis

```
SE_pbmc5k <- addGCBias(SE_pbmc5k, genome = BSgenome.Hsapiens.UCSC.hg19)
SE_pbmc5k_bg <- getBackgroundPeaks(SE_pbmc5k, niterations=200)
```

```
trait_import <- importBedScore(rowRanges(SE_pbmc5k), trait_file, colidx=5)
SE_pbmc5k_DEV <- computeWeightedDeviations(SE_pbmc5k, trait_import, background_peaks =
        SE_pbmc5k_bg)
```

Reformat results

```
z_score_mat <- data.frame(colData(SE_pbmc5k), z_score=t(assays(SE_pbmc5k_DEV)[["z"]]) %>% c)
head(z_score_mat)
```

```
##                                                names         x         y  color
## input1#GTCACGGAGCTCGGCT-1 input1#GTCACGGAGCTCGGCT-1  11.71388  1.903179 Mono-2
## input1#CTGAATGAGCAGAATT-1 input1#CTGAATGAGCAGAATT-1 -13.86186 -4.616170    T-1
## input1#CCTGCTACAATGGCAG-1 input1#CCTGCTACAATGGCAG-1  10.90323  1.913244 Mono-2
## input1#TCAGGTAAGAGCAGCT-1 input1#TCAGGTAAGAGCAGCT-1 -13.64482 -4.757390    T-1
## input1#GAGTGAGTCGGTCTCT-1 input1#GAGTGAGTCGGTCTCT-1  10.77266  1.872978 Mono-2
## input1#AGGCCCAAGTCTGCTA-1 input1#AGGCCCAAGTCTGCTA-1 -13.88653 -4.610587    T-1
##                           color2 sample cell_cluster    z_score
## input1#GTCACGGAGCTCGGCT-1     C5 input1           C5  0.3950389
## input1#CTGAATGAGCAGAATT-1     C1 input1           C1  0.0984394
## input1#CCTGCTACAATGGCAG-1     C5 input1           C5  0.3504030
## input1#TCAGGTAAGAGCAGCT-1     C1 input1           C1 -2.7724179
## input1#GAGTGAGTCGGTCTCT-1     C5 input1           C5 -0.4360599
## input1#AGGCCCAAGTCTGCTA-1     C1 input1           C1 -2.1425049
```

Generate the seed cell index (using the top 5% if too many cells are eligible)

```
seed_idx <- seedindex(z_score_mat$z_score, 0.05)
```

```
## Cells with enriched P < 0.05: 612
```

```
## Percent: 13%
```

```
## The top 5% of cells (N=228) were selected as seed cells
```

calculate scale factor

```
scale_factor <- cal_scalefactor(z_score=z_score_mat$z_score, 0.01)
```

```
## Scale factor is calculating from most enriched 1% of cells
```

Construct m-knn graph

*Calculate tfidf-mat

```
peak_by_cell_mat <- assay(SE_pbmc5k)
tfidf_mat <- tfidf(bmat=peak_by_cell_mat, mat_binary=TRUE, TF=TRUE, log_TF=TRUE)
```

```
## [info] binarize matrix
```

```
## [info] calculate tf
```

```
## [info] calculate idf
```

```
## [info] fast log tf-idf
```

Calculate lsi-mat

```
lsi_mat <- do_lsi(tfidf_mat, dims=30)
```

```
## SVD analysis of TF-IDF matrix
```

Calculate m-knn graph

```
mutualknn30 <- getmutualknn(lsi_mat, 30)
```

Network propagation

```
np_score <- randomWalk_sparse(intM=mutualknn30, queryGenes=rownames(mutualknn30)[seed_idx],
        gamma=0.05)
```

Trait relevant score (TRS) with scaled and normalized A few cells are singletons are removed from further analysis

```
omit_idx <- np_score==0
sum(omit_idx)
```

```
## [1] 23
```

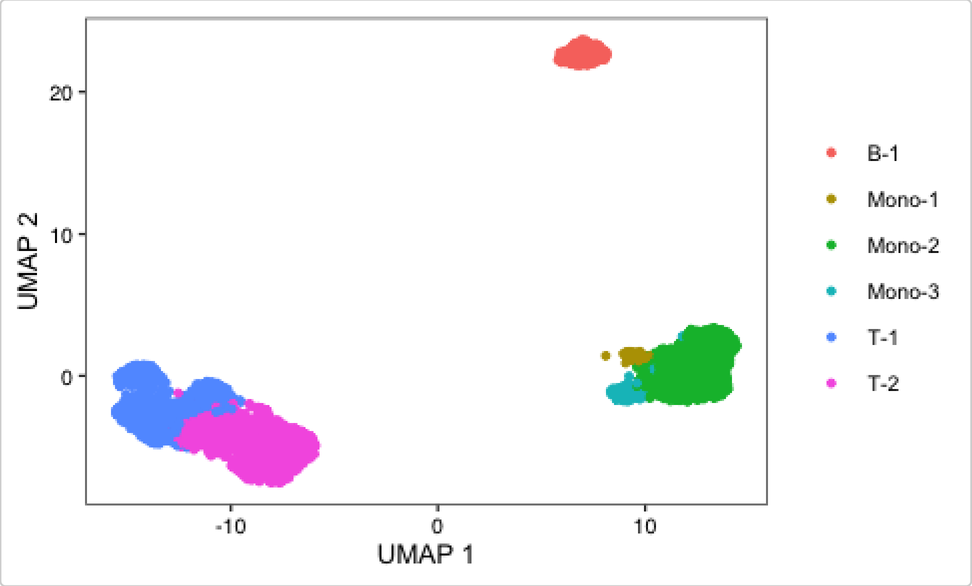
```
mutualknn30 <- mutualknn30[!omit_idx, !omit_idx]
np_score <- np_score[!omit_idx]
TRS <- np_score %>% capOutlierQuantile(., 0.95) %>% max_min_scale
TRS <- TRS * scale_factor
mono_mat <- data.frame(z_score_mat[!omit_idx, ], seed_idx[!omit_idx], np_score, TRS)
head(mono_mat)
```

```
##                                                names         x           y  color
## input1#GTCACGGAGCTCGGCT-1 input1#GTCACGGAGCTCGGCT-1  11.71388  1.903179 Mono-2
## input1#CTGAATGAGCAGAATT-1 input1#CTGAATGAGCAGAATT-1 -13.86186 -4.616170    T-1
## input1#CCTGCTACAATGGCAG-1 input1#CCTGCTACAATGGCAG-1  10.90323  1.913244 Mono-2
## input1#TCAGGTAAGAGCAGCT-1 input1#TCAGGTAAGAGCAGCT-1 -13.64482 -4.757390    T-1
## input1#GAGTGAGTCGGTCTCT-1 input1#GAGTGAGTCGGTCTCT-1  10.77266  1.872978 Mono-2
## input1#AGGCCCAAGTCTGCTA-1 input1#AGGCCCAAGTCTGCTA-1 -13.88653 -4.610587    T-1
##                            color2 sample cell_cluster    z_score
## input1#GTCACGGAGCTCGGCT-1     C5 input1           C5  0.3950389
## input1#CTGAATGAGCAGAATT-1     C1 input1           C1  0.0984394
## input1#CCTGCTACAATGGCAG-1     C5 input1           C5  0.3504030
## input1#TCAGGTAAGAGCAGCT-1     C1 input1           C1 -2.7724179
## input1#GAGTGAGTCGGTCTCT-1     C5 input1           C5 -0.4360599
## input1#AGGCCCAAGTCTGCTA-1     C1 input1           C1 -2.1425049
##                            seed_idx..omit_idx.     np_score         TRS
## input1#GTCACGGAGCTCGGCT-1                FALSE 3.804691e-05 0.213939514
## input1#CTGAATGAGCAGAATT-1                FALSE 2.209024e-07 0.001187911
## input1#CCTGCTACAATGGCAG-1                FALSE 6.088393e-05 0.342385858
## input1#TCAGGTAAGAGCAGCT-1                FALSE 2.220132e-07 0.001194159
## input1#GAGTGAGTCGGTCTCT-1                FALSE 4.785297e-05 0.269093513
## input1#AGGCCCAAGTCTGCTA-1                FALSE 2.572135e-07 0.001392142
```

UMAP plots of cell type annotation and cell-to-cell graph

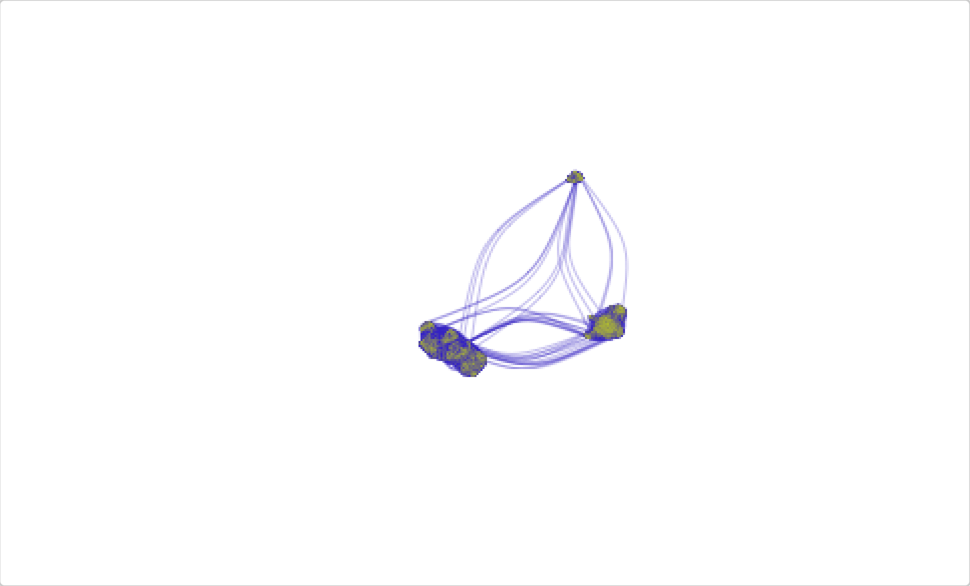
- **Cell type annotation**

```
p <- ggplot(data=mono_mat, aes(x, y, color=color)) + geom_point(size=1, na.rm = TRUE) +
    pretty_plot() + theme(legend.title = element_blank()) + xlab("UMAP 1") + ylab("UMAP 2")
p
```



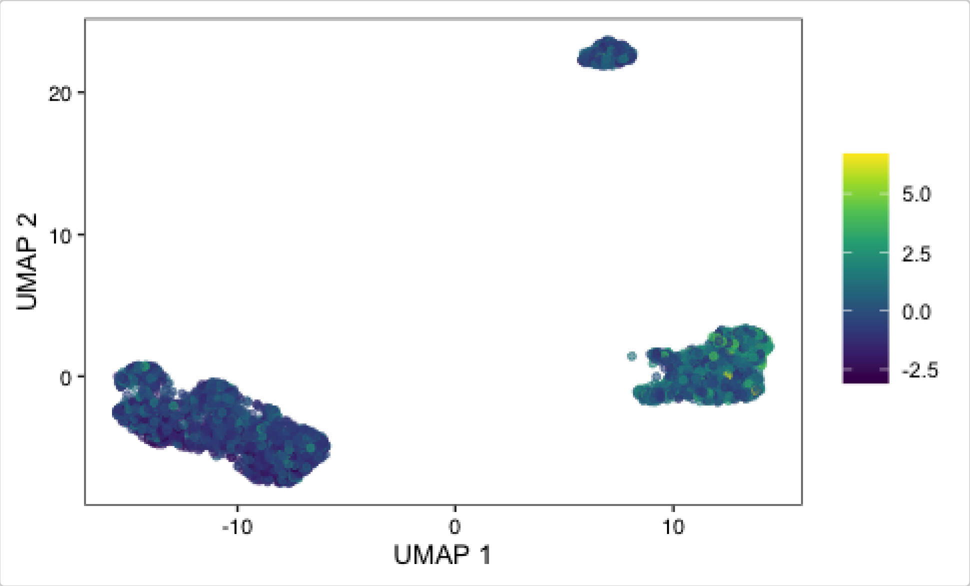
draw cell-to-cell graph if you have low-dimensional coordinates such as UMAP1 and UMAP2

```
mutualknn30_graph <- graph_from_adjacency_matrix(mutualknn30, mode = "undirected", diag = F)
plot.igraph(mutualknn30_graph, vertex.size=0.8, vertex.label=NA,
        vertex.color=adjustcolor("#c7ce3d", alpha.f = 1), vertex.frame.color=NA,
                edge.color=adjustcolor("#443dce", alpha.f = 1), edge.width=0.3, edge.curved=.5,
                layout=as.matrix(data.frame(mono_mat$x, mono_mat$y)))
```



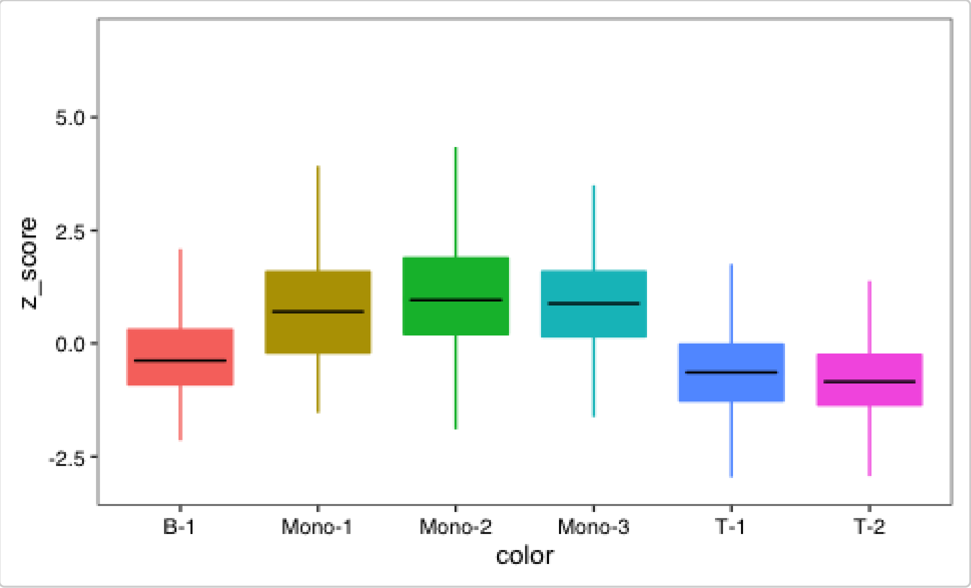
### Comparsion before and after SCAVENGE analysis - Z score based visualization **Scatter plot**

```
p1 <- ggplot(data=mono_mat, aes(x, y, color=z_score)) + geom_point(size=1, na.rm = TRUE,
        alpha = 0.6) +
scale_color_gradientn(colors = viridis) + scale_alpha()+
    pretty_plot() + theme(legend.title = element_blank()) + xlab("UMAP 1") + ylab("UMAP 2")
p1
```



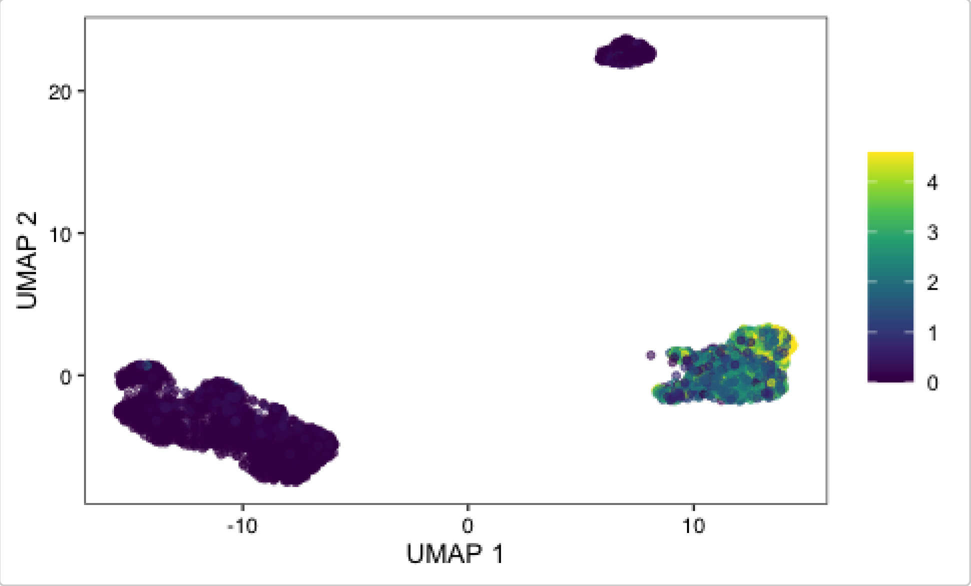
Bar plot

```
pp1 <- ggplot(data=mono_mat,  aes(x=color, y=z_score))  +
    geom_boxplot(aes(fill=color, color=color), outlier.shape=NA) +
    guides(fill=FALSE) + pretty_plot(fontsize = 10) +
    stat_summary(geom = "crossbar", width=0.65, fatten=0, color="black", fun.data =
        function(x){ return(c(y=median(x), ymin=median(x), ymax=median(x))) }) +
        theme(legend.position = "none")
pp1
```



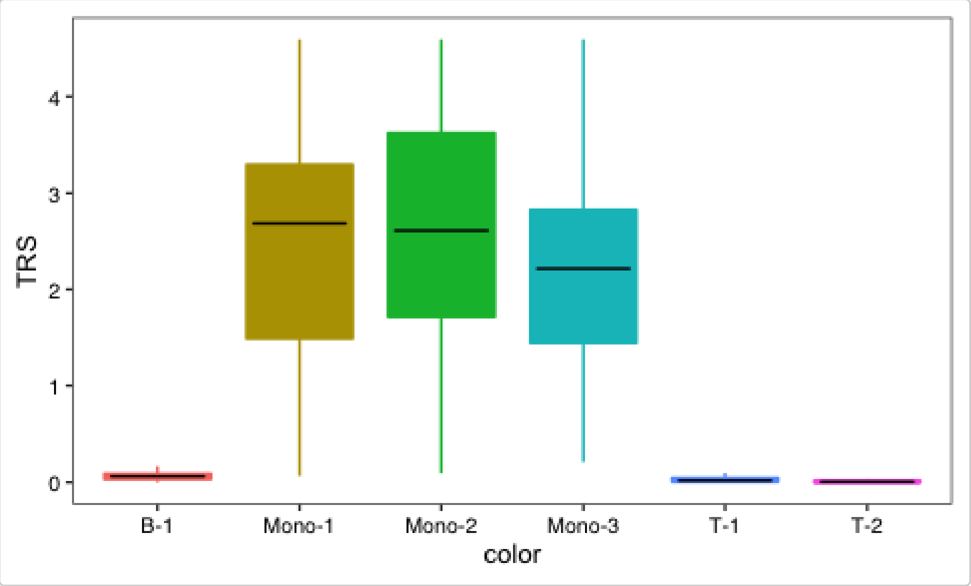
- SCAVENGE TRS based visualization **Scatter plot**

```
p2 <- ggplot(data=mono_mat, aes(x, y, color=TRS)) + geom_point(size=1, na.rm = TRUE, alpha =
        0.6) +
scale_color_gradientn(colors = viridis) + scale_alpha()+
    pretty_plot() + theme(legend.title = element_blank()) + xlab("UMAP 1") + ylab("UMAP 2")
p2
```



Bar plot

```
pp2 <- ggplot(data=mono_mat,  aes(x=color, y=TRS))  +
    geom_boxplot(aes(fill=color, color=color), outlier.shape=NA) +
    guides(fill=FALSE) + pretty_plot(fontsize = 10) +
    stat_summary(geom = "crossbar", width=0.65, fatten=0, color="black", fun.data =
        function(x){ return(c(y=median(x), ymin=median(x), ymax=median(x))) }) +
        theme(legend.position = "none")
pp2
```



Trait relevant cell determination from permutation test

About 2 mins please set @mycores >= 1 and @permutation_times >= 1,000 in the real setting

```
mono_permu <- get_sigcell_simple(knn_sparse_mat=mutualknn30, seed_idx=mono_mat$seed_idx,
        topseed_npscore=mono_mat$np_score, permutation_times=200,
        true_cell_significance=0.05, rda_output=F, mycores=4, rw_gamma=0.05)
```

```
## Get started!
```

```
## cells passed 0.001 threshold: 55.89%
```

```
## cells passed 0.01 threshold: 56.55%
```

```
## cells passed 0.05 threshold: 56.84%
```

```
## what proportion of seed cells over all cells: 5.02%
```

```
## your emprical P value threshold: 0.05
```

```
## what proportion of enriched cells over all cells: 43.16%
```

```
## what proportion of seed cells that are enriched cells: 100%
```

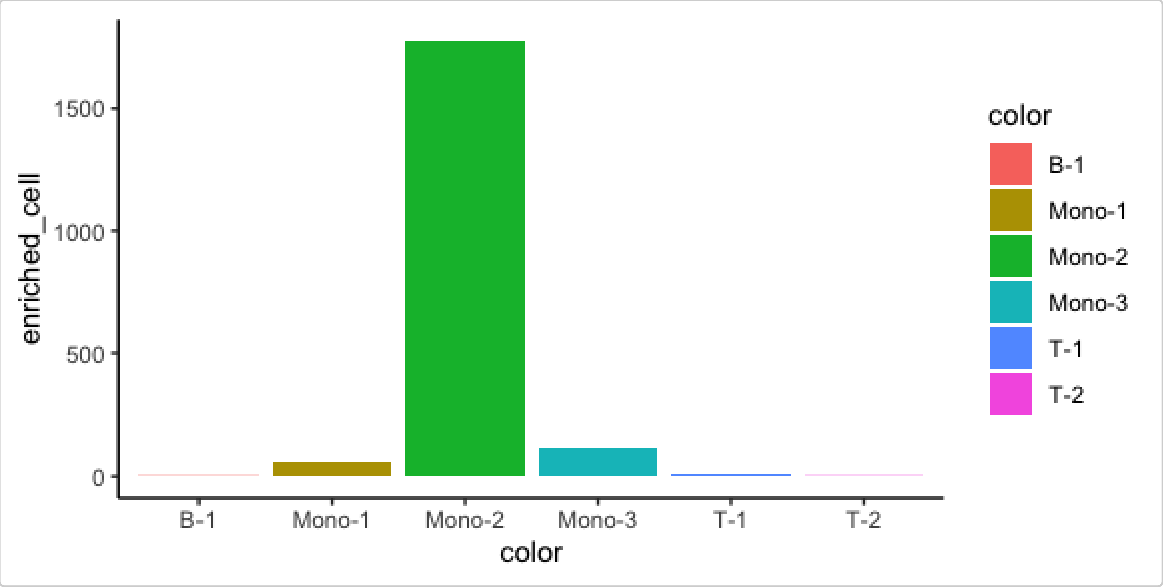
```
## fold of true cell over seed: 8.59
```

```
mono_mat2 <- data.frame(mono_mat, mono_permu)
```

Look at the distribution of statistically significant phenotypically enriched and depleted cells

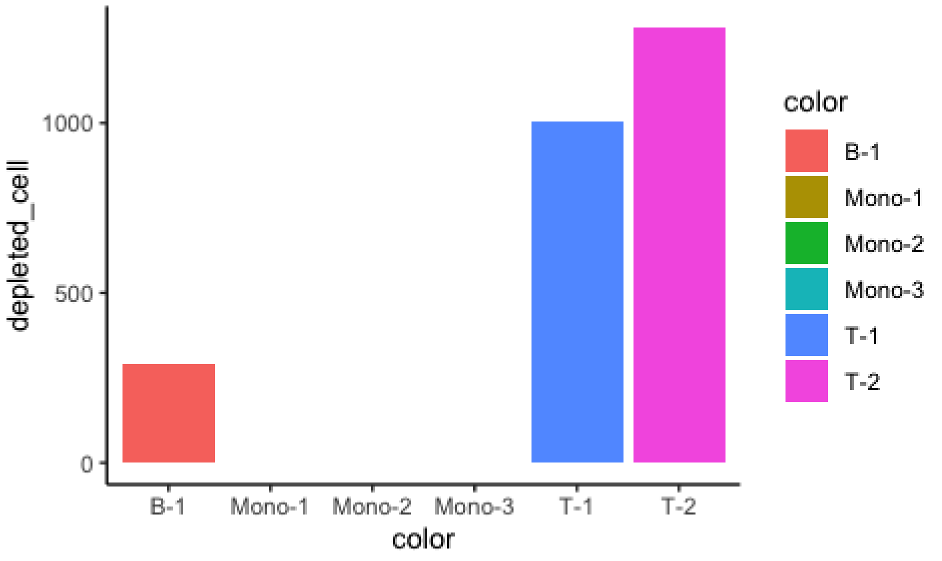
- Enriched cells

```
mono_mat2 %>%
    group_by(color) %>%
        summarise(enriched_cell=sum(true_cell_top_idx)) %>%
            ggplot(aes(x=color, y=enriched_cell, fill=color)) + geom_bar(stat="identity") +
        theme_classic()
```



- Depleted cells

```
mono_mat2$rev_true_cell_top_idx <- !mono_mat2$true_cell_top_idx
mono_mat2 %>%
    group_by(color) %>%
        summarise(depleted_cell=sum(rev_true_cell_top_idx)) %>%
            ggplot(aes(x=color, y=depleted_cell, fill=color)) + geom_bar(stat="identity") +
        theme_classic()
```


depleted_cell
1000
500
0
B-1
Mono-1
Mono-2
Mono-3
T-1
T-2
color
color
B-1
Mono-1
Mono-2
Mono-3
T-1
T-2