Step 2.2: Over-representation & gene set enrichment analysis

Carlos Gallardo & Christian Oertlin

17 April, 2023

```r
# Import libraries and helper functions
source("code/helper_functions.R")
library(tidyverse)
library(magrittr)
library(patchwork)
library(RColorBrewer)
library(qusage)
library(hypeR)
library(gage)
library(ComplexUpset)

# Colors
colPals <- vector(mode = "list")
colPals$time <- setNames(c("#FBAA3E", "#2C83BE", "#3EB6BD", "#A3D5B3", "#CD71A8"),
                         nm = c("day0", "day7", "day14", "day21", "day28"))
colPals$time_light <- setNames(c("#FDD6A1", "#A2CDE9", "#AFE2E5", "#DDF0E3", "#E8BDD6"),
                               nm = c("day0", "day7", "day14", "day21", "day28"))
colPals$time_dark <- setNames(c("#D87E04", "#174564", "#1F5C60", "#49A065", "#AA3C7E"),
                              nm = c("day0", "day7", "day14", "day21", "day28"))
colPals$inferno <- c("#000004", "#420A68", "#932667", "#DD513A", "#FCA50A", "#FCFFA4")
colPals$blood_cells <- setNames(c("#E54D34","#77A2D5","#B58B80"),
                                nm = c("granulocytes", "lymphocytes", "monocytes"))
colPals$cell_types <- setNames(c("#83D1F6","#FBAA3E","#FCCA7C","#B58B80","#E54D34",
                                 "#B3177E","#9A509F","#77A2D5","#CAC1DD","#36B449","#C1C1C1"),
                               nm = c("B cell", "Macrophage M1", "Macrophage M2",
                                      "Monocyte", "Neutrophil", "NK cell",
                                      "T cell CD4+ (non-regulatory)", "T cell CD8+",
                                      "T cell regulatory (Tregs)", "Myeloid dendritic cell",
                                      "uncharacterized cell"))
colPals$RdBu <- brewer.pal(11, name = "RdBu")
colPals$biotype <- setNames(c("#395982","#49BED9","#18A38A","#36B449","#826F99",
                              "#9852A5","#FBAA3E","#FCCA7C","#FCFFA4","#C1C1C1"),
                            nm = c("protein_coding", "lncRNA", "miRNA", "snoRNA",
                                   "IG_C_gene", "IG_V_gene", "TR_C_gene",
                                   "TR_J_gene", "TR_V_gene", "other"))
```

Load data

```r
# RNA-seq
RNAseq <- readRDS(file='data/rnaseq/rnaseq_volunteers_9&10_excl.rds')
DESeq2_DEGs <- readRDS(file='data/rnaseq/DESeq2_DEGs_unfilt_volunteers_9&10_excl.rds')
DESeq2_DEGs_filt <- readRDS(file='data/rnaseq/DESeq2_DEGs_filt_volunteers_9&10_excl.rds')
DEGs_clusters <- readRDS(file='data/rnaseq/DEGs_kmeans_clusters.rds')

# Gene Sets
gene_sets <- list(
  # GO (2021/09/10) http://current.geneontology.org/products/pages/downloads.html
  gobp = readGMT('data/resources/gobp_human_sep2021.gmt'),
  gomf = readGMT('data/resources/gomf_human_sep2021.gmt'),
  gocc = readGMT('data/resources/gocc_human_sep2021.gmt'),
  # Reactome (2021/09/10) https://reactome.org/download-data
  reactome = readGMT('data/resources/reactome_human_sep2021.gmt'),
  # MSigDB v7.4 (2021/11/19) https://www.gsea-msigdb.org/gsea/msigdb/human/collections.jsp
  msig_immune = readGMT('data/resources/c7.immunesigdb.v7.4.symbols.gmt')
)
```

Over-representation analysis (ORA)

```
# get unique gene names per cluster
DEG_sets <- list(
  cluster_1_up = DEGs_clusters$DEGs_clusters_FC %>%
    filter(Cluster == '1') %>%
    pull(GeneSymbol) %>%
    gsub('(.+)_\\d+','\\1',.) %>%
    str_subset(., pattern = '^ENS', negate = T) %>%
    unique(),
  cluster_2_down = DEGs_clusters$DEGs_clusters_FC %>%
    filter(Cluster == '2') %>%
    pull(GeneSymbol) %>%
    gsub('(.+)_\\d+','\\1',.) %>%
    str_subset(., pattern = '^ENS', negate = T) %>%
    unique()
)

# get gene background
gene_background <- RNAseq$filt$annotation$GeneSymbol %>%
  gsub('(.+)_\\d+','\\1',.) %>%
  str_subset(., pattern = '^ENS', negate = T) %>%
  unique()

# store over-representation analysis results
hyper_res <- list()
res_ORA <- list()
```

GO Biological Process (GO_BP)

```
# GO Biological Process
hyper_res$gobp <- hypeR::hypeR(signature = DEG_sets,
                               genesets = gene_sets$gobp$genesets,
                               test = 'hypergeometric',
                               background = gene_background)

# append descriptions and filter
res_ORA$gobp <- lapply(setNames(names(DEG_sets),names(DEG_sets)), function(x) {
  hyper_res$gobp$data[[x]]$data %>%
    mutate(description=recode(label,
                              !!!setNames(gene_sets$gobp$geneset.descriptions,
                                          gene_sets$gobp$geneset.names))) %>%
    mutate(cluster = x,
           gene_set = 'GO_BP') %>%
    filter(fdr<0.05)
}) %>% bind_rows()

paste("GO Biological Process (FDR<0.05):", nrow(res_ORA$gobp))
```

```
## [1] "GO Biological Process (FDR<0.05): 5"
```

GO Molecular Function (GO_MF)

```
# GO Molecular Function
hyper_res$gomf <- hypeR::hypeR(signature = DEG_sets,
                               genesets = gene_sets$gomf$genesets,
                               test = 'hypergeometric',
                               background = gene_background)

# append descriptions and filter
res_ORA$gomf <- lapply(setNames(names(DEG_sets),names(DEG_sets)), function(x) {
  hyper_res$gomf$data[[x]]$data %>%
    mutate(description=recode(label,
                              !!!setNames(gene_sets$gomf$geneset.descriptions,
                                          gene_sets$gomf$geneset.names))) %>%
    mutate(cluster = x,
           gene_set = 'GO_MF') %>%
    filter(fdr<0.05)
}) %>% bind_rows()

paste("GO Molecular Function (FDR<0.05):", nrow(res_ORA$gomf))
```

```
## [1] "GO Molecular Function (FDR<0.05): 20"
```

GO Cellular Component (GO_CC)

```
# GO Cellular Component
hyper_res$gocc <- hypeR::hypeR(signature = DEG_sets,
                               genesets = gene_sets$gocc$genesets,
                               test = 'hypergeometric',
                               background = gene_background)

# append descriptions and filter
res_ORA$gocc <- lapply(setNames(names(DEG_sets),names(DEG_sets)), function(x) {
  hyper_res$gocc$data[[x]]$data %>%
    mutate(description=recode(label,
                              !!!setNames(gene_sets$gocc$geneset.descriptions,
                                          gene_sets$gocc$geneset.names))) %>%
    mutate(cluster = x,
           gene_set = 'GO_CC') %>%
    filter(fdr<0.05)
}) %>% bind_rows()

paste("GO Cellular Component (FDR<0.05):", nrow(res_ORA$gocc))
```

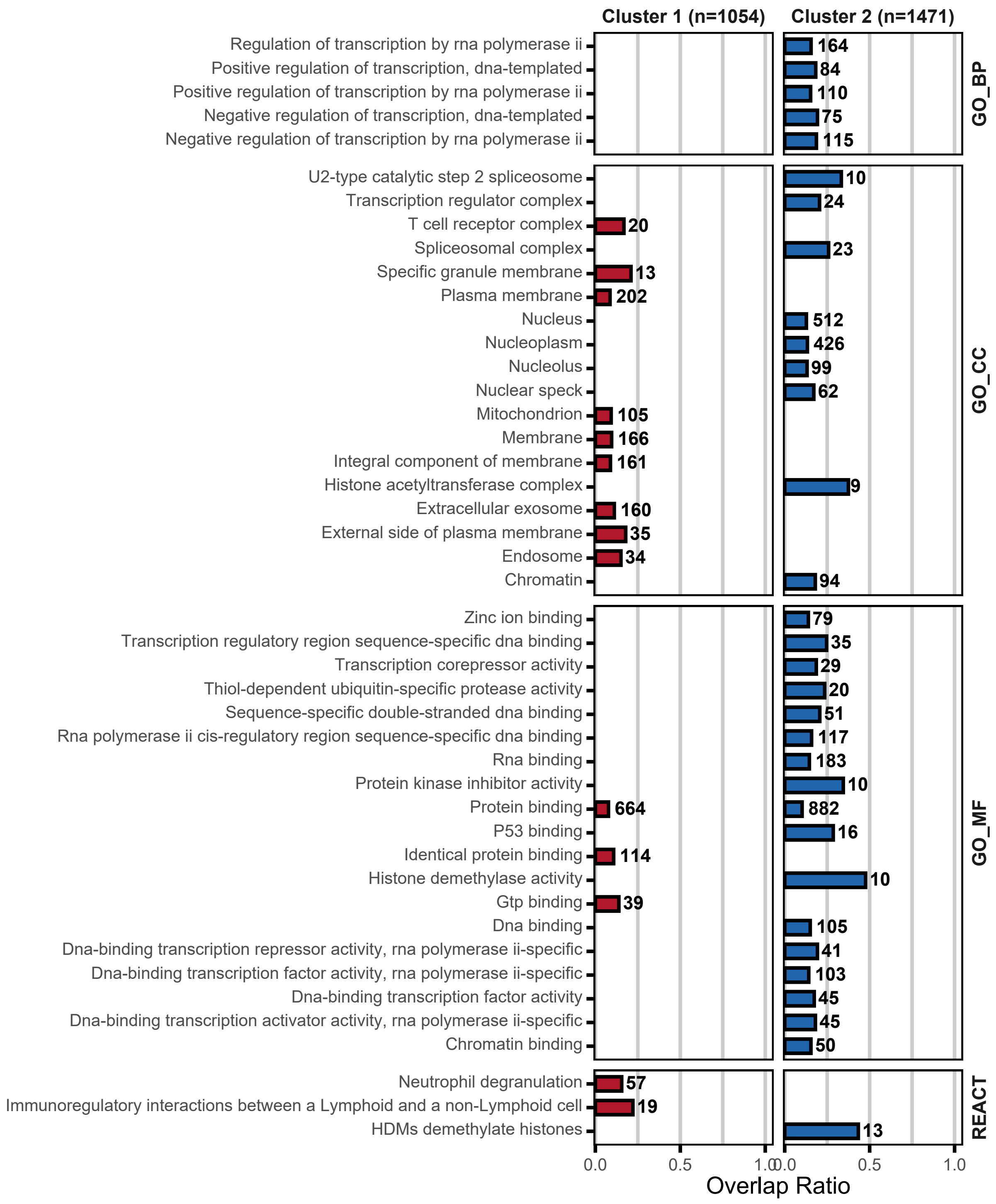
```
## [1] "GO Cellular Component (FDR<0.05): 18"
```

Reactome Pathways (REACT)

```
# Reactome Pathways
hyper_res$reactome <- hypeR::hypeR(signature = DEG_sets,
                               genesets = gene_sets$reactome$genesets,
                               test = 'hypergeometric',
                               background = gene_background)

# append descriptions and filter
res_ORA$reactome <- lapply(setNames(names(DEG_sets),names(DEG_sets)), function(x) {
  hyper_res$reactome$data[[x]]$data %>%
    mutate(description=recode(label,
                              !!!setNames(gene_sets$reactome$geneset.descriptions,
                                          gene_sets$reactome$geneset.names))) %>%
    mutate(cluster = x,
           gene_set = 'REACT') %>%
    filter(fdr<0.05)
}) %>% bind_rows()

paste("Reactome Pathways (FDR<0.05):", nrow(res_ORA$reactome))
```

```
## [1] "Reactome Pathways (FDR<0.05): 3"
```

```
res_ORA_all <- res_ORA %>%
  bind_rows() %>%
  mutate(overlap_ratio = overlap/geneset,
         cluster = recode(cluster, !!!setNames(c('1', '2'), names(DEG_sets)))) %>%
  select(gene_set, cluster, label, description, geneset,
         overlap, overlap_ratio, pval, fdr, hits)

lbls <- c("1" = paste0("Cluster 1 (n=",sum(DEGs_clusters$DEGs_clust_expr$cluster=='1'),")"),
          "2" = paste0("Cluster 2 (n=",sum(DEGs_clusters$DEGs_clust_expr$cluster=='2'),")"),
          "GO_BP" = "GO_BP",
          "GO_MF" = "GO_MF",
          "GO_CC" = "GO_CC",
          "REACT" = "REACT")

ggplot(res_ORA_all, aes(x=overlap_ratio, y=description, fill=cluster, label=overlap)) +
  geom_bar(stat='identity', size=1, width=0.6, color='black') +
  geom_text(color = 'black', angle = 0, hjust=-0.2, vjust=0.4, fontface='bold') +
  scale_x_continuous(breaks = c(0,0.5,1),
                     limits = c(0,1),
                     expand = expansion(mult = c(.01, .05))) +
  scale_fill_manual(values = colPals$RdBu[c(2,10)]) +
  facet_grid(gene_set~cluster, scales="free_y", space = "free_y", labeller = as_labeller(lbls)) +
  xlab('Overlap Ratio') +
  ylab('') +
  theme_bw() +
  theme(
    text = element_text(family = 'Arial', size = 14),
    axis.text.x.bottom = element_text(size = 10, hjust = 0.5, vjust = 0.5),
    axis.text.y.left = element_text(size = 10, hjust = 1, vjust = 0.3),
    panel.border = element_rect(color = "black", fill = NA, size = 1),
    axis.ticks = element_line(color = "black", size = 1),
    axis.ticks.length = unit(1.5, 'mm'),
    panel.grid.major.x = element_line(color = "grey80", linetype = "solid", size = 1),
    panel.grid.minor.x = element_line(color = "grey80", linetype = "solid", size = 1),
    panel.grid.major.y = element_blank(),
    panel.grid.minor.y = element_blank(),
    legend.position = 'none',
    strip.background = element_blank(),
    strip.text = element_text(face = "bold"),
    strip.text.y = element_text(angle = 90)
  )
```



```r
ggsave(filename = "plots/figS6_DEGs_clusters_ORA_results.pdf", width = 8.5, height = 10, units = "in", dpi = 300, device = cairo_pdf)
```

Gene set enrichment analysis (GSEA)

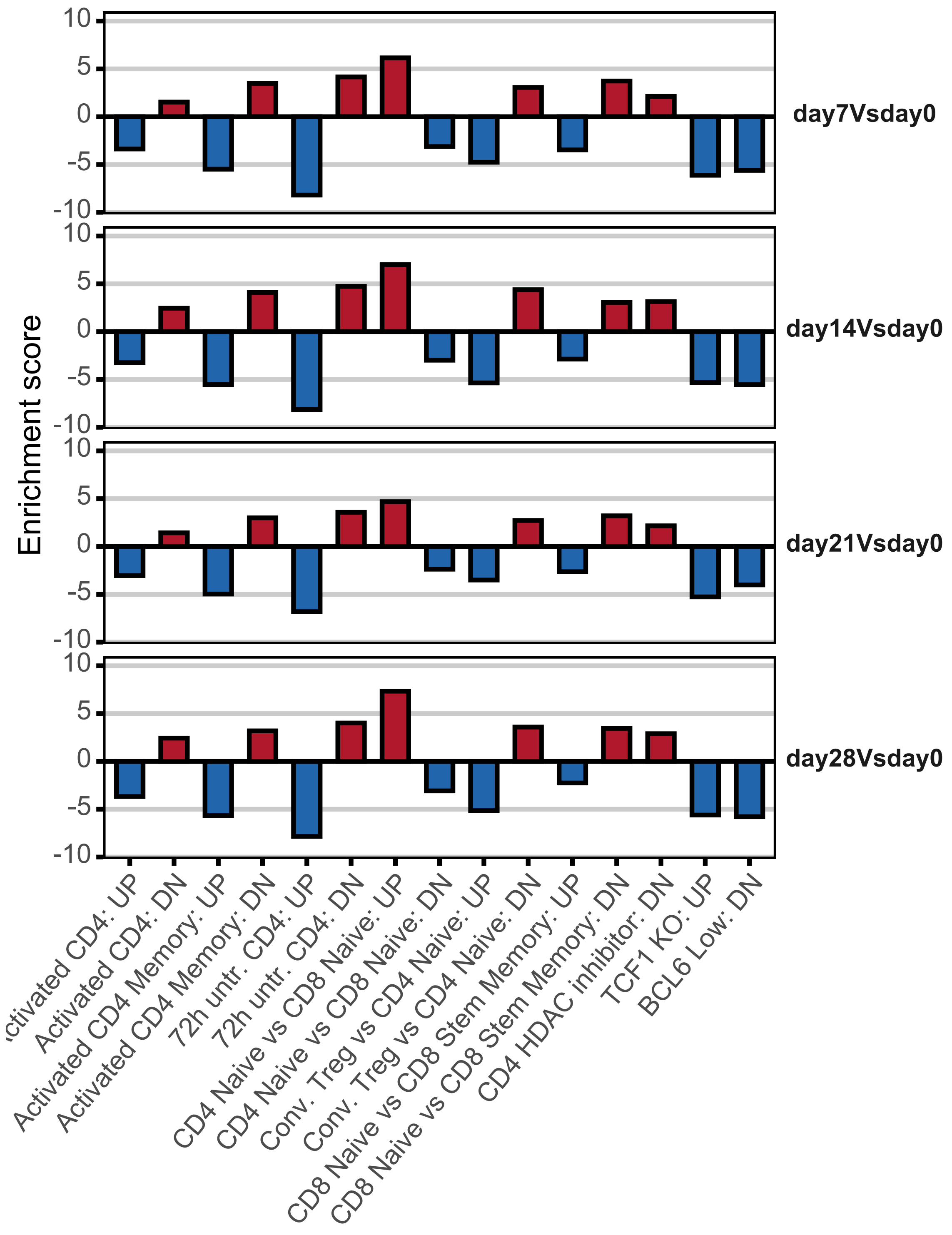
```r
# run GSEA on immune gene sets from MSigDB (Broad Institute)
# filter gene sets with little relevance to isolated T cells (i.e., B cells, etc.)
gene_sets$msig_immune$genesets <- gene_sets$msig_immune$genesets[grepl("TCELL",names(gene_sets$msig_immune$genesets))]
gene_sets$msig_immune$genesets <- gene_sets$msig_immune$genesets[!grepl("NK",names(gene_sets$msig_immune$genesets))]
gene_sets$msig_immune$genesets <- gene_sets$msig_immune$genesets[!grepl("MAST",names(gene_sets$msig_immune$genesets))]
gene_sets$msig_immune$genesets <- gene_sets$msig_immune$genesets[!grepl("BCELL",names(gene_sets$msig_immune$genesets))]
gene_sets$msig_immune$genesets <- gene_sets$msig_immune$genesets[!grepl("MONOCYTE",names(gene_sets$msig_immune$genesets))]
gene_sets$msig_immune$genesets <- gene_sets$msig_immune$genesets[!grepl("NEUTROPHIL",names(gene_sets$msig_immune$genesets))]
gene_sets$msig_immune$genesets <- gene_sets$msig_immune$genesets[!grepl("EOSINOPHIL",names(gene_sets$msig_immune$genesets))]
gene_sets$msig_immune$genesets <- gene_sets$msig_immune$genesets[!grepl("DC",names(gene_sets$msig_immune$genesets))]
gene_sets$msig_immune$genesets <- gene_sets$msig_immune$genesets[!grepl("HIV",names(gene_sets$msig_immune$genesets))]
tokeep <- gene_sets$msig_immune$geneset.names %in% names(gene_sets$msig_immune$genesets)
gene_sets$msig_immune$geneset.names <- gene_sets$msig_immune$geneset.names[tokeep]
gene_sets$msig_immune$geneset.descriptions <- gene_sets$msig_immune$geneset.descriptions[tokeep]


# rank genes by different time point comparisons
gene_ranks <- lapply(DESeq2_DEGs[2:length(DESeq2_DEGs)], function(x) {
  df <- x %>%
    mutate(GeneSymbol = gsub('(.+)_\\d+','\\1',.$GeneSymbol)) %>%
    filter(!duplicated(GeneSymbol)) %>%
    mutate(rank_score = -log10(padj) * sign(log2FoldChange)) %>%
    arrange(desc(rank_score))
  rank_score <- setNames(df$rank_score, df$GeneSymbol)
  rank(rank_score)
})

gage_GSEA <- runGage(gene_ranks, gene_sets$msig_immune$genesets, cutOff = 0.05)
```

```
## [1] "gs.data needs to be a matrix-like object!"
## [1] "gs.data needs to be a matrix-like object!"
## [1] "gs.data needs to be a matrix-like object!"
## [1] "there are 30 signficantly up-regulated gene sets"
## [1] "there are 59 signficantly down-regulated gene sets"
## [1] "gs.data needs to be a matrix-like object!"
## [1] "gs.data needs to be a matrix-like object!"
## [1] "gs.data needs to be a matrix-like object!"
## [1] "there are 90 signficantly up-regulated gene sets"
## [1] "there are 50 signficantly down-regulated gene sets"
## [1] "gs.data needs to be a matrix-like object!"
## [1] "gs.data needs to be a matrix-like object!"
## [1] "gs.data needs to be a matrix-like object!"
## [1] "there are 24 signficantly up-regulated gene sets"
## [1] "there are 52 signficantly down-regulated gene sets"
## [1] "gs.data needs to be a matrix-like object!"
## [1] "gs.data needs to be a matrix-like object!"
## [1] "gs.data needs to be a matrix-like object!"
## [1] "there are 40 signficantly up-regulated gene sets"
## [1] "there are 56 signficantly down-regulated gene sets"
```

```r
res_GSEA_sig <- lapply(setNames(names(gage_GSEA),names(gage_GSEA)), function(x) {
  greater <- gage_GSEA[[x]]$sig$greater %>%
    as.data.frame() %>%
    rownames_to_column(var = 'description') %>%
    mutate(group = 'greater',
           comparison = x)
  less <- gage_GSEA[[x]]$sig$less %>%
    as.data.frame() %>%
    rownames_to_column(var = 'description') %>%
    mutate(group = 'less',
           comparison = x)
  bind_rows(greater, less)
}) %>% bind_rows()

# curate and group relevant immune sets that are enriched
immune_set_groups <- res_GSEA_sig %>%
  arrange(q.val) %>%
  filter(!duplicated(description)) %>%
  select(description) %>%
  mutate(label = '') %>%
  # untreated CD4 T cells
  mutate(label = ifelse(grepl('72H',description) & grepl('UNTREATED',description) &
                          grepl('CD4',description) & grepl('UP',description),
                         '72h untr. CD4: UP', label)) %>%
  mutate(label = ifelse(grepl('72H',description) & grepl('UNTREATED',description) &
                           grepl('CD4',description) & grepl('DN',description),
                         '72h untr. CD4: DN', label)) %>%
  # activated memory CD4 T cells
  mutate(label = ifelse(grepl('ACT',description) & grepl('MEMORY',description) &
                           grepl('CD4',description) & grepl('UP',description),
                         'Activated CD4 Memory: UP', label)) %>%
  mutate(label = ifelse(grepl('ACT',description) & grepl('MEMORY',description) &
                           grepl('CD4',description) & grepl('DN',description),
                         'Activated CD4 Memory: DN', label)) %>%
  # activated CD4 T cells
  mutate(label = ifelse(grepl('ACT',description) & !grepl('MEMORY|TH2|TH1',description) &
                           grepl('CD4',description) & grepl('UP',description),
                         'Activated CD4: UP', label)) %>%
  mutate(label = ifelse(grepl('ACT',description) & !grepl('MEMORY|TH2|TH1',description) &
                           grepl('CD4',description) & grepl('DN',description),
                         'Activated CD4: DN', label)) %>%
  # conventional T regs vs CD4 naive
  mutate(label = ifelse(grepl('NAIVE_CD4',description) & grepl('CONV_TREG',description) &
                           grepl('UP',description),
                         'Conv. Treg vs CD4 Naive: UP', label)) %>%
  mutate(label = ifelse(grepl('NAIVE_CD4',description) & grepl('CONV_TREG',description) &
                           grepl('DN',description),
                         'Conv. Treg vs CD4 Naive: DN', label)) %>%
  # CD4 vs CD8 naive
  mutate(label = ifelse(grepl('CD8_VS_CD4_NAIVE',description) & grepl('UP',description),
                         'CD4 Naive vs CD8 Naive: UP', label)) %>%
  mutate(label = ifelse(grepl('CD8_VS_CD4_NAIVE',description) & grepl('DN',description),
                         'CD4 Naive vs CD8 Naive: DN', label)) %>%
  # CD8 naive vs CD8 stem memory
  mutate(label = ifelse(grepl('CD8_STEM_CELL_MEMORY_VS_NAIVE_CD8',description) & grepl('UP',description),
                         'CD8 Naive vs CD8 Stem Memory: UP', label)) %>%
  mutate(label = ifelse(grepl('CD8_STEM_CELL_MEMORY_VS_NAIVE_CD8',description) & grepl('DN',description),
                         'CD8 Naive vs CD8 Stem Memory: DN', label)) %>%
  # CD4 HDAC inhibitor-treated
  mutate(label = ifelse(grepl('HDAC_INHIBITOR_TREATED_CD4',description) & grepl('DN',description),
                         'CD4 HDAC inhibitor: DN', label)) %>%
  # TCF1 KO
  mutate(label = ifelse(grepl('WT_VS_TCF1_KO_MEMORY_CD8_TCELL_UP',description),
                         'TCF1 KO: UP', label)) %>%
  # BCL6 Low
  mutate(label = ifelse(grepl('BCL6_HIGH_TFH_VS_TFH_CD4_TCELL_DN',description),
                         'BCL6 Low: DN', label)) %>%
  filter(!label == '')

# intersect gene sets to gene background
immune_sets_background <- lapply(gene_sets$msig_immune$genesets, function(x) {
  genes_in_background <- intersect(x, gene_background)
  genes_in_background
})

# get enrichment scores for all enriched sets in all comparisons
res_GSEA_all_selected <- lapply(setNames(names(gage_GSEA),names(gage_GSEA)), function(x) {

  greater <- gage_GSEA[[x]]$all$greater %>%
    as.data.frame() %>%
    rownames_to_column(var = 'description') %>%
    filter(description %in% immune_set_groups$description & stat.mean > 0) %>%
    mutate(DEG_overlap = lapply(immune_sets_background[description], function(x){
      length(intersect(x, unlist(DEG_sets)))
    }) %>% unlist() %>% as.numeric()) %>%
    mutate(overlap_ratio = DEG_overlap/set.size) %>%
    mutate(group = 'greater',
           comparison = x,
           label = recode(description, !!!setNames(immune_set_groups$label,
                                                    nm = immune_set_groups$description)),
           hits = lapply(immune_sets_background[description], function(x){
             paste(x, collapse = ',')
           }) %>% unlist() %>% as.character())

  less <- gage_GSEA[[x]]$all$less %>%
    as.data.frame() %>%
    rownames_to_column(var = 'description') %>%
    filter(description %in% immune_set_groups$description & stat.mean < 0) %>%
    mutate(DEG_overlap = lapply(immune_sets_background[description], function(x){
      length(intersect(x, unlist(DEG_sets)))
    }) %>% unlist() %>% as.numeric()) %>%
    mutate(overlap_ratio = DEG_overlap/set.size) %>%
    mutate(group = 'less',
           comparison = x,
           label = recode(description, !!!setNames(immune_set_groups$label,
                                                   nm = immune_set_groups$description)),
           hits = lapply(immune_sets_background[description], function(x){
             paste(x, collapse = ',')
           }) %>% unlist() %>% as.character())

  bind_rows(greater, less)

}) %>%
  bind_rows() %>%
  dplyr::rename(p_geomean = p.geomean, enrichment_score = stat.mean, pval = p.val, qval = q.val,
                set_size = set.size) %>%
  select(label, description, comparison, group, p_geomean, enrichment_score,
         set_size, DEG_overlap, overlap_ratio, exp1, pval, qval, hits)

# collapse immune set groups and average enrichment scores by comparison
immune_set_group_lbls <- c(
  'Activated CD4: UP', 'Activated CD4: DN', 'Activated CD4 Memory: UP', 'Activated CD4 Memory: DN',
  '72h untr. CD4: UP', '72h untr. CD4: DN', 'CD4 Naive vs CD8 Naive: UP', 'CD4 Naive vs CD8 Naive: DN',
  'Conv. Treg vs CD4 Naive: UP', 'Conv. Treg vs CD4 Naive: DN', 'CD8 Naive vs CD8 Stem Memory: UP',
  'CD8 Naive vs CD8 Stem Memory: DN', 'CD4 HDAC inhibitor: DN', 'TCF1 KO: UP', 'BCL6 Low: DN'
)
res_GSEA_collapsed <- lapply(setNames(immune_set_group_lbls,immune_set_group_lbls), function(i){
  df <- res_GSEA_all_selected %>%
    filter(label == i) %>%
    group_by(label, comparison)

  df %>%
    summarise(enrichment_score_mean = mean(enrichment_score),
              pval_mean = mean(pval),
              qval_mean = mean(qval)) %>%
    mutate(hits = lapply(df$hits, function(x){
      strsplit(x, split = ',')
    }) %>% unlist() %>% unique() %>% paste(collapse = ','))
}) %>%
  bind_rows() %>%
  mutate(set_size = lapply(hits, function(x) {
    strsplit(x, ',')[[1]] %>% length()
  }) %>% unlist() %>% as.numeric(),
  DEG_overlap = lapply(hits, function(x){
    strsplit(x, ',')[[1]] %>% intersect(., unlist(DEG_sets)) %>% length()
  }) %>% unlist() %>% as.numeric()) %>%
  mutate(overlap_ratio = DEG_overlap/set_size) %>%
  select(label, comparison, enrichment_score_mean, set_size, DEG_overlap,
         overlap_ratio, pval_mean, qval_mean, hits) %>%
  mutate(label = factor(label, levels = immune_set_group_lbls),
         comparison = factor(comparison, levels = c('day7Vsday0','day14Vsday0',
                                                    'day21Vsday0','day28Vsday0'))) %>%
  arrange(label, comparison)
```

Average enrichment of collapsed immune sets

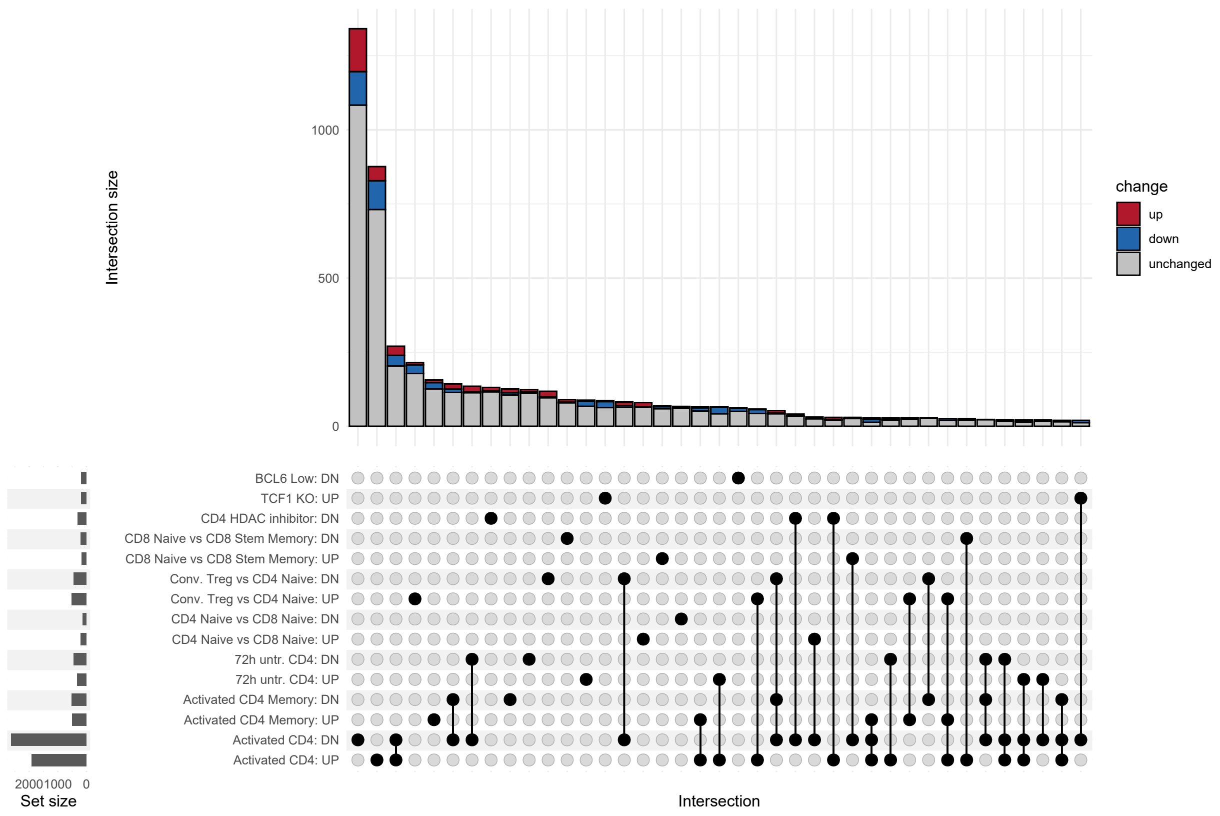
```
df <- res_GSEA_collapsed %>%
  mutate(type = ifelse(enrichment_score_mean > 0, 'up', 'down')) %>%
  mutate(type = factor(type, levels = c('up', 'down')))

ggplot(df, aes(x=label, y=enrichment_score_mean, fill=type)) +
  geom_bar(stat='identity', size=1, width=0.6, color='black') +
  geom_hline(yintercept = 0, color='black', size=1) +
  scale_y_continuous(breaks = c(-10,-5,0,5,10),
                     limits = c(-10,10),
                     expand = expansion(mult = c(.01, .05))) +
  scale_fill_manual(values = colPals$RdBu[c(2,10)]) +
  facet_wrap(~comparison, ncol=1, strip.position='right') +
  xlab('') +
  ylab('Enrichment score') +
  theme_bw() +
  theme(
    text = element_text(family = 'Arial', size = 14),
    axis.text.x.bottom = element_text(angle=50, size = 12, hjust = 1, vjust = 1),
    axis.text.y.left = element_text(size = 12, hjust = 1, vjust = 0.3),
    panel.border = element_rect(color = "black", fill = NA, size = 1),
    axis.ticks = element_line(color = "black", size = 1),
    axis.ticks.length = unit(1.1, 'mm'),
    panel.grid.major.y = element_line(color = "grey80", linetype = "solid", size = 1),
    panel.grid.minor.y = element_blank(),
    panel.grid.major.x = element_blank(),
    panel.grid.minor.x = element_blank(),
    legend.position = 'none',
    strip.background = element_blank(),
    strip.text = element_text(face = "bold"),
    strip.text.y = element_text(angle = 0)
  )
```



```
ggsave(filename = "plots/fig2C_GSEA_immune_gene_sets.pdf", width = 6, height = 9, units = "in", dpi = 300, device = cairo_pdf)
```

Gene overlap between immune sets

```
immune_set_all_genes <- lapply(setNames(immune_set_group_lbls,immune_set_group_lbls), function(x){
  genes <- res_GSEA_collapsed %>%
    filter(label == x) %>%
    pull(hits) %>%
    unique()
  strsplit(genes, split = ',')[[1]]
}) %>% unlist() %>% unique()

gene_expr_change <- DESeq2_DEGs$day14Vsday0 %>%
  mutate(GeneSymbol = gsub('(.+)_\\d+','\\1',GeneSymbol)) %>%
  filter(GeneSymbol %in% immune_set_all_genes) %>%
  filter(!duplicated(GeneSymbol)) %>%
  mutate(change = 'unchanged') %>%
  mutate(change = ifelse(GeneSymbol %in% unlist(DEG_sets) & log2FoldChange > 0, 'up', change)) %>%
  mutate(change = ifelse(GeneSymbol %in% unlist(DEG_sets) & log2FoldChange < 0, 'down', change))

gene_expr_change <- setNames(gene_expr_change$change,
                             nm = gene_expr_change$GeneSymbol)

df <- lapply(setNames(immune_set_group_lbls,immune_set_group_lbls), function(x){
  genes <- res_GSEA_collapsed %>%
    filter(label == x) %>%
    pull(hits) %>%
    unique()
  genes <- strsplit(genes, split = ',')[[1]]
  immune_set_all_genes %in% genes
}) %>%
  bind_cols() %>%
  mutate(genes = immune_set_all_genes) %>%
  mutate(change = recode(genes, !!!gene_expr_change)) %>%
  column_to_rownames(var = 'genes') %>%
  mutate(change = factor(change, levels = c('up', 'down', 'unchanged')))

upset(df,
      immune_set_group_lbls,
      min_size = 20,
      set_sizes=upset_set_size(position = 'left'),
      base_annotations=list(
        'Intersection size'=intersection_size(
          counts=F,
          mapping=aes(fill=change),
          color="black",
          size=0.5
        ) + scale_fill_manual(values = c(colPals$RdBu[c(2,10)], '#C1C1C1'))
      ),
      name='Intersection',
      width_ratio=0.1,
      height_ratio=0.7,
      sort_sets=FALSE)
```



```
ggsave(filename = "plots/figS8_GSEA_immune_gene_sets_overlap.pdf", width = 12, height = 6, units = "in", dpi = 300, device = cairo_pdf)
```

Exports

```
gene_set_analysis_res <- list(ORA_results = res_ORA_all,
                              GSEA_immune = res_GSEA_all_selected,
                              GSEA_immune_collapsed = res_GSEA_collapsed)

openxlsx::write.xlsx(
  gene_set_analysis_res,
  file = "tables/dataS5_gene_set_analysis.xlsx",
  rowNames=F,
  overwrite=T
)
```

SessionInfo

```
sessionInfo()
```

```
## R version 4.2.1 (2022-06-23 ucrt)
## Platform: x86_64-w64-mingw32/x64 (64-bit)
## Running under: Windows 10 x64 (build 19044)
##
## Matrix products: default
##
## locale:
## [1] LC_COLLATE=English_United States.utf8
## [2] LC_CTYPE=English_United States.utf8
## [3] LC_MONETARY=English_United States.utf8
## [4] LC_NUMERIC=C
## [5] LC_TIME=English_United States.utf8
##
## attached base packages:
## [1] stats     graphics  grDevices utils     datasets  methods   base
##
```

```
## other attached packages:
##  [1] ComplexUpset_1.3.5 gage_2.48.0        hypeR_1.14.0       qusage_2.32.0
##  [5] limma_3.54.2       RColorBrewer_1.1-3 patchwork_1.1.2    magrittr_2.0.3
##  [9] forcats_1.0.0      stringr_1.5.0      dplyr_1.1.1        purrr_1.0.1
## [13] readr_2.1.4        tidyr_1.3.0        tibble_3.2.1       ggplot2_3.4.2
## [17] tidyverse_1.3.2
##
## loaded via a namespace (and not attached):
##   [1] googledrive_2.1.0      colorspace_2.1-0       ellipsis_0.3.2
##   [4] estimability_1.4.1     XVector_0.38.0         fs_1.6.1
##   [7] rstudioapi_0.14        farver_2.1.1           bit64_4.0.5
##  [10] AnnotationDbi_1.60.2   fansi_1.0.4            mvtnorm_1.1-3
##  [13] lubridate_1.9.2        xml2_1.3.3             cachem_1.0.7
##  [16] knitr_1.42             polyclip_1.10-4        jsonlite_1.8.4
##  [19] broom_1.0.4            GO.db_3.16.0           dbplyr_2.3.2
##  [22] png_0.1-8              graph_1.76.0           ggforce_0.4.1
##  [25] shiny_1.7.4            msigdbr_7.5.1          compiler_4.2.1
##  [28] httr_1.4.5             emmeans_1.8.5          backports_1.4.1
##  [31] Matrix_1.5-3           fastmap_1.1.1          gargle_1.3.0
##  [34] cli_3.6.1              later_1.3.0            tweenr_2.0.2
##  [37] visNetwork_2.1.2       htmltools_0.5.5        tools_4.2.1
##  [40] igraph_1.4.1           GenomeInfoDbData_1.2.9 gtable_0.3.3
##  [43] glue_1.6.2             reshape2_1.4.4         Rcpp_1.0.10
##  [46] Biobase_2.58.0         cellranger_1.1.0       vctrs_0.6.1
##  [49] Biostrings_2.66.0      babelgene_22.9         svglite_2.1.1
##  [52] nlme_3.1-157           xfun_0.38              openxlsx_4.2.5.1
##  [55] rvest_1.0.3            timechange_0.2.0       mime_0.12
##  [58] lifecycle_1.0.3        googlesheets4_1.1.0    zlibbioc_1.44.0
##  [61] MASS_7.3-57            scales_1.2.1           hms_1.1.3
##  [64] promises_1.2.0.1       yaml_2.3.7             memoise_2.0.1
##  [67] stringi_1.7.12         RSQLite_2.3.0          highr_0.10
##  [70] S4Vectors_0.36.2       BiocGenerics_0.44.0    zip_2.2.2
##  [73] GenomeInfoDb_1.34.9    bitops_1.0-7           rlang_1.1.0
##  [76] pkgconfig_2.0.3        systemfonts_1.0.4      evaluate_0.20
##  [79] lattice_0.20-45        labeling_0.4.2         htmlwidgets_1.6.2
##  [82] bit_4.0.5              tidyselect_1.2.0       plyr_1.8.8
##  [85] R6_2.5.1               IRanges_2.32.0         fftw_1.0-7
##  [88] generics_0.1.3         DBI_1.1.3              pillar_1.9.0
##  [91] haven_2.5.2            withr_2.5.0            RCurl_1.98-1.12
##  [94] KEGGREST_1.38.0        reactable_0.4.4        modelr_0.1.11
##  [97] crayon_1.5.2           utf8_1.2.3             tzdb_0.3.0
## [100] rmarkdown_2.21         grid_4.2.1             readxl_1.4.2
## [103] blob_1.2.4             reprex_2.0.2           digest_0.6.31
## [106] webshot_0.5.4          xtable_1.8-4           httpuv_1.6.9
## [109] stats4_4.2.1           munsell_0.5.0          viridisLite_0.4.1
## [112] kableExtra_1.3.4
```