1. Merging_cleaning_heatmap_gene_expression

Fay

2022-09-15

GAPDH HKG
B-actin HKG
Ppia HKG
Ppip HKG
CDC42 HKG susceptible to DNA contamination

Housekeeping genes selected: GAPDH and PPIB

Relm-b mucosal defense factor (goblet cells)
Muc2 the major secretory mucin within the gastrointestinal tract
TFF3 mucosal defense factor (goblet cells)
Muc5ac similar to MUC2, produced by surface goblet cells
NKp46 NK marker
F4/80 macropahge marker (distinguish by immune response trend)
Mpo myeloperoxidase in Neutrophils
MyD88 TLR protein, NF-kB IRAK protein, inflammation marker by TLR MyD88-Dependent Pathway caspase-1 inflammasome marker (IL-1b and IL-18 production)
IL-1Ra natuaral IL-1b antagonist for infection control (if not increase in Tregs is seen)
CXCL9, immune cell migration marker + Th1 activator (confirm FACS)
CXCR3, CXCL9 and CXCL11 receptor
IL-6 TNF inhibitor,
IL-12ra T-cell marker Th1
IFN-y compare with IFN-y producing cells and IFN-y ELISAs, should correlate wit PRF1, NKp46 and F4/80. One of these cell types just have to be doing the job!
IRG6A autonomous cell defense (opsonization)
TNF-a upregulated in eimeria but not well explained. Could be present and driving infection where IFN-y isn't
IL-17 in case IFN-y isn't coming up but pathogenicity is
TRIF Type I IFN production TRIF Dependent Pathway
Socs1 JAK/STAT signaling pathway, proinflammatory regulating + T-cell differentiation, could explain severity
IDO1 DC, monocyte and MC protein regulating T-cell activity
Prf1 perforin, should be dominant in primary infections, but must be correlated between T-cell and NK cell expresions
CD56 CD56bright = more cytokine producing NKs, CD56dim = more direct cytotoxic killing
IL-4
IL-13
IL-10

Load libraries

Import data

Challenge = experimental challenge infection data

SOTA = State of the Art, of wild data

```
Challenge <- read.csv("input_data/Challenge_infections.csv")


SOTA <- read.csv("input_data/SOTA_Data_Product.csv")

# Vectors for selecting genes

#Lab genes
# The measurements of IL.12 and IRG6 are done with an other assay and will
#ignore for now
Gene_lab   <- c("IFNy", "CXCR3", "IL.6", "IL.13", "IL.10",
                "IL1RN","CASP1", "CXCL9", "IDO1", "IRGM1", "MPO",
                "MUC2", "MUC5AC", "MYD88", "NCR1", "PRF1", "RETNLB", "SOCS1",
                "TICAM1", "TNF") #"IL.12", "IRG6")

Genes_wild   <- c("IFNy", "CXCR3", "IL.6", "IL.13", "IL.10",
                  "IL1RN","CASP1", "CXCL9", "IDO1", "IRGM1", "MPO",
                  "MUC2", "MUC5AC", "MYD88", "NCR1", "PRF1", "RETNLB", "SOCS1",
                  "TICAM1", "TNF") #, "IL.12", "IRG6")

Facs_lab <- c("Position", "CD4", "Treg", "Div_Treg", "Treg17", "Th1",
                    "Div_Th1", "Th17", "Div_Th17", "CD8", "Act_CD8",
                    "Div_Act_CD8", "IFNy_CD4", "IFNy_CD8","Treg_prop",
                    "IL17A_CD4")

Facs_wild <- c( "Treg", "CD4", "Treg17", "Th1", "Th17", "CD8",
                     "Act_CD8", "IFNy_CD4", "IL17A_CD4", "IFNy_CD8")
```

Cleaning

Challenge

```
## Adding missing grouping variables: `infection`
## Joining, by = c("infection", "EH_ID")
```

Join wild and lab data

```
## [1] 32
```

```
## [1] 158
```

```
## [1] 186
```

Lab

Heatmap on lab gene expression: data preparation

```
gene <- as_tibble(data) %>%
    dplyr::filter(origin == "Lab", infection == "challenge", dpi == dpi_max) %>%
    dplyr::group_by(Mouse_ID) %>%
    dplyr::select(c(Mouse_ID, MC.Eimeria, all_of(Gene_lab)))

gene <- unique(gene)


# turn the data frame into a matrix and transpose it. We want to have each cell
 # type as a row name
 gene <- t(as.matrix(gene %>%
                         dplyr::select(c(Mouse_ID, all_of(Gene_lab)))))

 #switch the matrix back to a data frame format
 gene <- as.data.frame(gene)

 # turn the first row into column names
 gene %>%
     row_to_names(row_number = 1) -> heatmap_data

 table(rowSums(is.na(heatmap_data)) == nrow(heatmap_data))
```

```
##
## FALSE  TRUE
##    17     3
```

```
 # turn the columns to numeric other wise the heatmap function will not work
 heatmap_data[] <- lapply(heatmap_data, function(x) as.numeric(as.character(x)))

 # remove columns with only NAs
 heatmap_data <- Filter(function(x)!all(is.na(x)), heatmap_data)

 #remove rows with only Nas
 heatmap_data <-  heatmap_data[, colSums(is.na(heatmap_data)) !=
                                nrow(heatmap_data)]


#Prepare the annotation data frame
annotation_df <- as_tibble(data) %>%
    dplyr::filter(origin == "Lab", infection == "challenge", dpi == dpi_max)%>%
    dplyr::group_by(Mouse_ID) %>%
    dplyr::select(c("Mouse_ID", "Parasite_challenge", "infection_history",
                    "mouse_strain", "max_WL"))

annotation_df <- unique(annotation_df) %>%
    dplyr::filter(Mouse_ID %in% colnames(heatmap_data))

annotation_df <- as.data.frame(annotation_df)
```

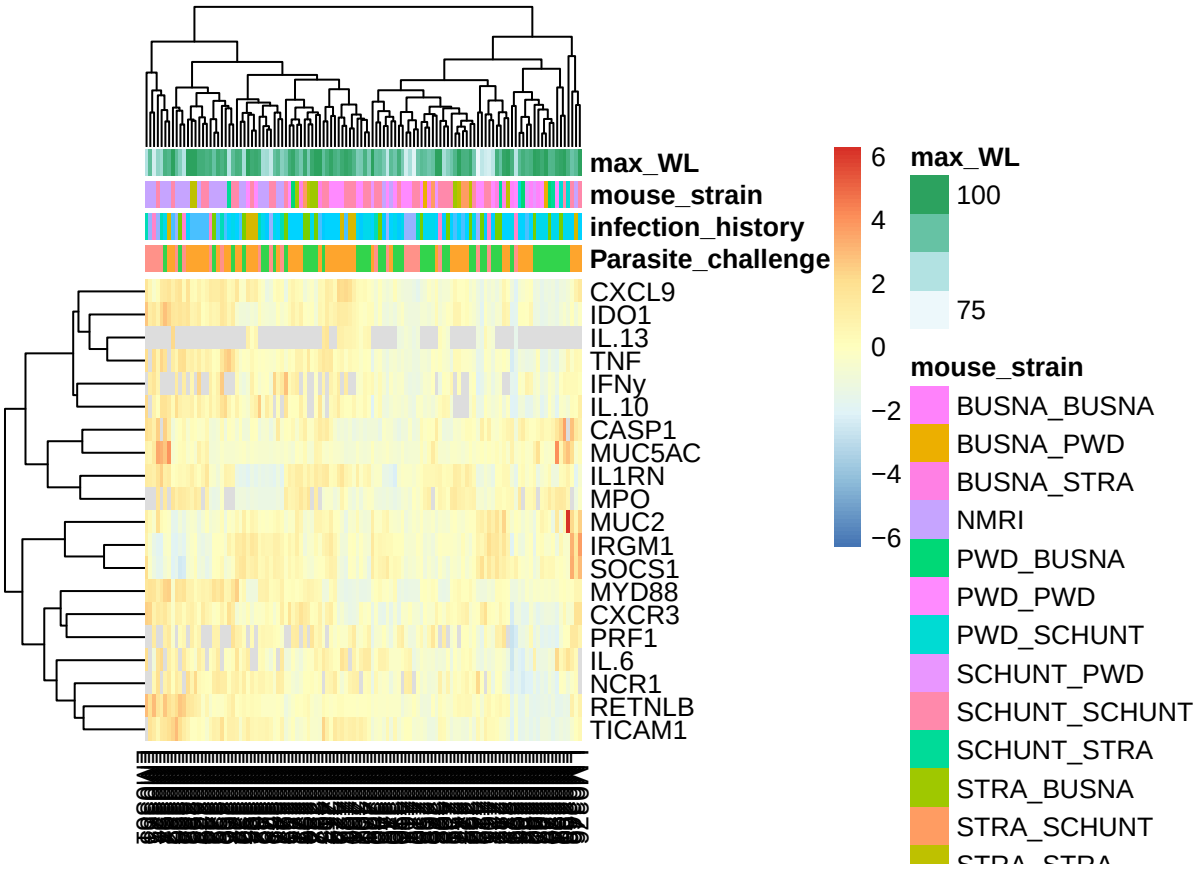
```
 ### Prepare the annotation columns for the heatmap
rownames(annotation_df) <- annotation_df$Mouse_ID

# Match the row names to the heatmap data frame
rownames(annotation_df) <- colnames(heatmap_data)

#remove the unecessary column
annotation_df <- annotation_df %>% dplyr::select(-Mouse_ID, )

heatmap_lab <- heatmap_data
```

Heatmap on lab gene expression data: plot



Field

Heatmap on Field gene expression: data preparation

```
gene <- as_tibble(data) %>%
    dplyr::filter(origin == "Field") %>%
    dplyr::group_by(Mouse_ID) %>%
  drop_na(HI) %>%
```

```
    dplyr::select(c(Mouse_ID, all_of(Genes_wild)))

gene <- unique(gene)

# I will remove nas. See towards the end of this chunk why I did it in this way
gene <- gene %>% drop_na(MYD88)


#
# turn the data frame into a matrix and transpose it. We want to have each cell
 # type as a row name
 gene <- t(as.matrix(gene))

 #switch the matrix back to a data frame format
 gene <- as.data.frame(gene)

 # turn the first row into column names
 gene %>%
     row_to_names(row_number = 1) -> heatmap_data

 table(rowSums(is.na(heatmap_data)) == nrow(heatmap_data))
```

```
##
## FALSE
##    20
```

```
 # turn the columns to numeric other wise the heatmap function will not work
 heatmap_data[] <- lapply(heatmap_data, function(x) as.numeric(as.character(x)))

 # remove columns with only NAs
 heatmap_data <- Filter(function(x)!all(is.na(x)), heatmap_data)

 #remove rows with only Nas
 heatmap_data <-  heatmap_data[, colSums(is.na(heatmap_data)) !=
                                nrow(heatmap_data)]


#Prepare the annotation data frame
annotation_df <- as_tibble(data) %>%
    dplyr::filter(origin == "Field")%>%
    dplyr::group_by(Mouse_ID) %>%
    dplyr::select(c("Mouse_ID", "Sex", "HI")) %>%
  drop_na()

annotation_df <- unique(annotation_df) %>%
    dplyr::filter(Mouse_ID %in% colnames(heatmap_data))


annotation_df <- as.data.frame(annotation_df)



 ### Prepare the annotation columns for the heatmap
```

```
rownames(annotation_df) <- annotation_df$Mouse_ID

# Match the row names to the heatmap data frame
rownames(annotation_df) <- colnames(heatmap_data)

#remove the unecessary column
annotation_df <- annotation_df %>% dplyr::select(-Mouse_ID, )

# heatmap gives error: Error in hclust(d, method = method) :
#  NA/NaN/Inf in foreign function call (arg 10)
#Let's figure this out
g_field <- data %>%
    dplyr::filter(origin == "Field") %>%
    dplyr::group_by(Mouse_ID) %>%
    dplyr::select(all_of(Genes_wild))
```

```
## Adding missing grouping variables: `Mouse_ID`
```

```
# How many nas?
is.na(g_field) %>% table()
```

```
## .
## FALSE  TRUE
##  7269 33072
```

```
is.infinite(unlist(g_field)) %>% table() #there are no infinite values
```

```
## .
## FALSE
## 40341
```

```
dim(g_field)
```

```
## [1] 1921   21
```

```
is.na(annotation_df) %>% table() # there is an na value in the annotation
```

```
## .
## FALSE
##   630
```

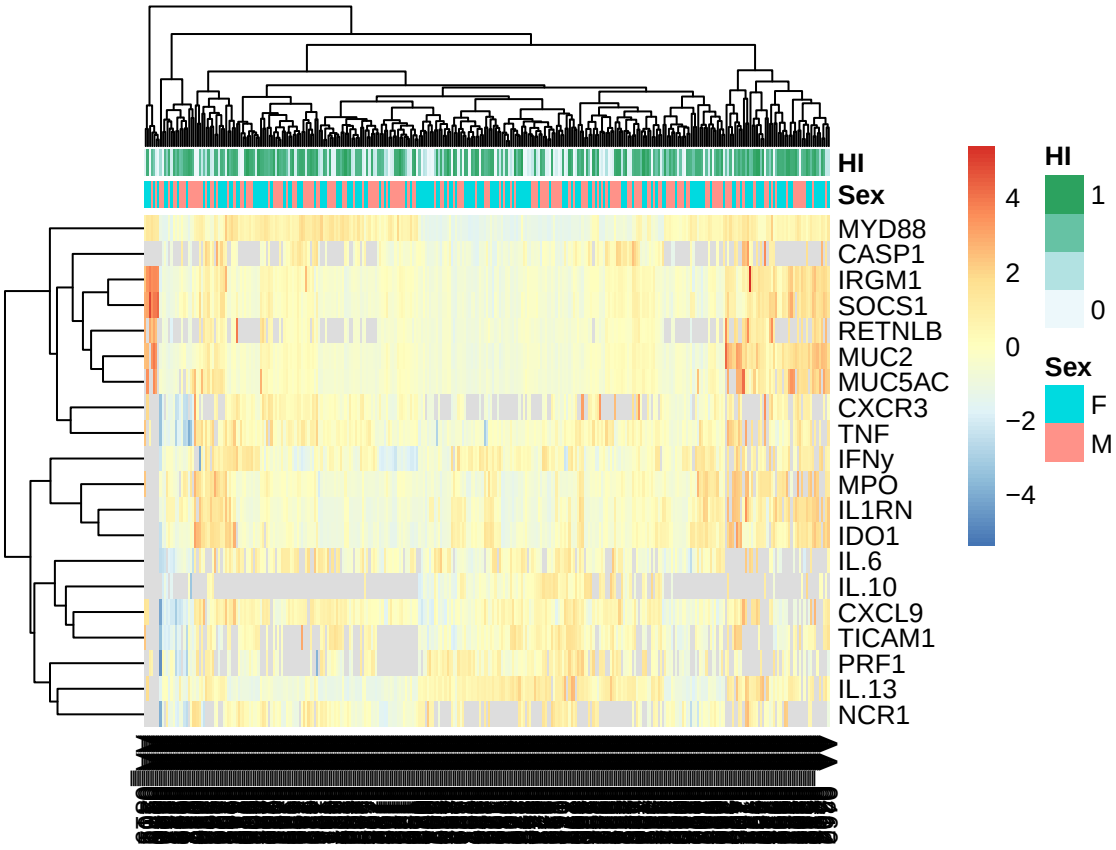
```
# df! that is what is causing the clustering error probably

annotation_df <- annotation_df %>% drop_na()

heatmap_data <- heatmap_data %>%
  dplyr::select(row.names(annotation_df))

# I will have to reduce the nas in order to get some clustering going on
# MYD88 is the most complete gene. I will remove the nas from the nas rows of
#this genes further up the analysis and see if it works out
heatmap_field <- heatmap_data
```

Heatmap on lab gene expression data: plot



Lab + Field Heatmap combination

```
heatmap_lab_field <- cbind(heatmap_lab, heatmap_field)


#Prepare the annotation data frame
annotation_df <- as_tibble(data) %>%
    dplyr::select(origin, Mouse_ID)

annotation_df <- unique(annotation_df) %>%
    dplyr::filter(Mouse_ID %in% colnames(heatmap_lab_field))

annotation_df <- as.data.frame(annotation_df)

 ### Prepare the annotation columns for the heatmap
rownames(annotation_df) <- annotation_df$Mouse_ID

# Match the row names to the heatmap data frame
rownames(annotation_df) <- colnames(heatmap_lab_field)

#remove the unecessary column
annotation_df <- annotation_df %>% dplyr::select(-Mouse_ID, )
```

Lab - Field data heatmap - plot

```
pheatmap(heatmap_lab_field, annotation_col = annotation_df, scale = "row")
```