Gene normalization

Fay

2022-11-04

Run before imputation!

Aim:

Normalize gene expression

I have tried the delta delta ct to get the fold gene expression and the delta ct method.

As we are having data from wild infections with unknown status we can't do the delta delta ct method. For that we would have needed control samples - for example uninfected vs infected. I tried to correct for that by getting the mean of every sample for each target gene and using it as a control but that didn't work. The values became non-comparable and then my data became very bad at predicting. By using the delta ct method, I increased the accuracy of my predictions.

Since the data sets are "mixed", wild and laboratory data, plus differnt runs, the delta ct should be enough to get the difference between the housekeeping gene and each sample (representing each run of the sample).

Another problem is that we have measurements for the house keeping genes PPIB and GAPDH. For the field samples, more than 50 % of PPIB is missing, whereas for the laboratory samples 80 % of GAPDH is missing.

I would have liked to use the geometric mean of two housekeeping genes. In this case I am limited to use one for each data set.

Immune Gene Expression Normalization

Gene expression data obtained from our quantitative PCR (qPCR) experiments were normalized and processed through a series of steps to ensure validity and reliability of our findings.

First, we normalized our target gene (TAR) expression levels to a reference gene (REF), a process known as delta cycle threshold (DeltaCq) calculation. This was done to control for variations between samples that may have arisen due to differences in PCR efficiency or loading discrepancies (refer to a paper on your reference gene selection and the DeltaCq method, e.g., Vandesompele et al., 2002).

After normalization to the reference gene, we then transformed the DeltaCq values using an exponential function (2^(-DeltaCq)), a process known as exponential expression transform. This transformation allowed us to convert the values into relative expression levels, providing a clear representation of gene expression fold changes (Livak and Schmittgen, 2001).

We then averaged the technical replicates for each sample and calculated the standard deviation (again, reference a paper explaining why this is a standard practice, possibly a statistical paper or a similar study in your field).

Next, we normalized the relative expression values to our treatment control group. The specific details of this normalization step varied depending on the design of our experiments and the specific comparisons we wished to make (reference a paper that describes normalization in your experimental context).

Finally, we calculated the percentage of knockdown (% KD) using the relative expression values (DeltaDeltaCq). The % KD quantifies the extent of gene knockdown or inhibition (refer to a paper explaining the concept and calculation of % KD).

https://www.ncbi.nlm.nih.gov/pmc/articles/PMC3489534/#:~:text=We%20suggest%20that%20normalization%20be,imputati
In publication: We suggest that normalization be done first followed by missing value imputations. Software tools such as DanteR as well as stand-alone functions in R and Matlab may be used to perform normalization, imputation, significance analysis and visualization.

Load libraries

```
library(mice)
```

```
##
## Attaching package: 'mice'

## The following object is masked from 'package:stats':
##
##     filter

## The following objects are masked from 'package:base':
##
##     cbind, rbind
```

```
library(tidyr)
library(tidyverse)
```

```
## -- Attaching packages --------------------------------------- tidyverse 1.3.2 --

## v ggplot2 3.4.1      v dplyr   1.0.10
## v tibble  3.2.1      v stringr 1.5.0
## v readr   2.1.3      v forcats 0.5.2
## v purrr   0.3.5
## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter() masks mice::filter(), stats::filter()
## x dplyr::lag()    masks stats::lag()
```

```
library(VIM)
```

```
## Loading required package: colorspace
## Loading required package: grid
## VIM is ready to use.
##
## Suggestions and bug-reports can be submitted at: https://github.com/statistikat/VIM/issues
##
## Attaching package: 'VIM'
##
## The following object is masked from 'package:datasets':
##
##     sleep
```

```
library(fitdistrplus)
```

```
## Loading required package: MASS
##
## Attaching package: 'MASS'
##
```

```
## The following object is masked from 'package:dplyr':
##
##     select
##
## Loading required package: survival
```

```
library(fitur)
```

```
##
## Attaching package: 'fitur'
##
## The following object is masked from 'package:purrr':
##
##     rdunif
```

```
library(visdat)
library(DESeq2)
```

```
## Loading required package: S4Vectors
## Loading required package: stats4
## Loading required package: BiocGenerics
##
## Attaching package: 'BiocGenerics'
##
## The following objects are masked from 'package:dplyr':
##
##     combine, intersect, setdiff, union
##
## The following objects are masked from 'package:mice':
##
##     cbind, rbind
##
## The following objects are masked from 'package:stats':
##
##     IQR, mad, sd, var, xtabs
##
## The following objects are masked from 'package:base':
##
##     anyDuplicated, append, as.data.frame, basename, cbind, colnames,
##     dirname, do.call, duplicated, eval, evalq, Filter, Find, get, grep,
##     grepl, intersect, is.unsorted, lapply, Map, mapply, match, mget,
##     order, paste, pmax, pmax.int, pmin, pmin.int, Position, rank,
##     rbind, Reduce, rownames, sapply, setdiff, sort, table, tapply,
##     union, unique, unsplit, which.max, which.min
##
##
## Attaching package: 'S4Vectors'
##
## The following objects are masked from 'package:dplyr':
##
##     first, rename
##
## The following object is masked from 'package:tidyr':
##
##     expand
```

```
##
## The following objects are masked from 'package:base':
##
##     expand.grid, I, unname
##
## Loading required package: IRanges
##
## Attaching package: 'IRanges'
##
## The following objects are masked from 'package:dplyr':
##
##     collapse, desc, slice
##
## The following object is masked from 'package:purrr':
##
##     reduce
##
## Loading required package: GenomicRanges
## Loading required package: GenomeInfoDb
## Loading required package: SummarizedExperiment
## Loading required package: MatrixGenerics
## Loading required package: matrixStats
##
## Attaching package: 'matrixStats'
##
## The following object is masked from 'package:dplyr':
##
##     count
##
##
## Attaching package: 'MatrixGenerics'
##
## The following objects are masked from 'package:matrixStats':
##
##     colAlls, colAnyNAs, colAnys, colAvgsPerRowSet, colCollapse,
##     colCounts, colCummaxs, colCummins, colCumprods, colCumsums,
##     colDiffs, colIQRDiffs, colIQRs, colLogSumExps, colMadDiffs,
##     colMads, colMaxs, colMeans2, colMedians, colMins, colOrderStats,
##     colProds, colQuantiles, colRanges, colRanks, colSdDiffs, colSds,
##     colSums2, colTabulates, colVarDiffs, colVars, colWeightedMads,
##     colWeightedMeans, colWeightedMedians, colWeightedSds,
##     colWeightedVars, rowAlls, rowAnyNAs, rowAnys, rowAvgsPerColSet,
##     rowCollapse, rowCounts, rowCummaxs, rowCummins, rowCumprods,
##     rowCumsums, rowDiffs, rowIQRDiffs, rowIQRs, rowLogSumExps,
##     rowMadDiffs, rowMads, rowMaxs, rowMeans2, rowMedians, rowMins,
##     rowOrderStats, rowProds, rowQuantiles, rowRanges, rowRanks,
##     rowSdDiffs, rowSds, rowSums2, rowTabulates, rowVarDiffs, rowVars,
##     rowWeightedMads, rowWeightedMeans, rowWeightedMedians,
##     rowWeightedSds, rowWeightedVars
##
## Loading required package: Biobase
## Welcome to Bioconductor
##
##     Vignettes contain introductory material; view with
##     'browseVignettes()'. To cite Bioconductor, see
##     'citation("Biobase")', and for packages 'citation("pkgname")'.
##
##
## Attaching package: 'Biobase'
##
## The following object is masked from 'package:MatrixGenerics':
##
##     rowMedians
##
## The following objects are masked from 'package:matrixStats':
##
##     anyMissing, rowMedians
```

Load data

Import data

```
  hm <- read.csv("output_data/1.MICE_cleaned_data.csv")
```

I only include GAPDH as a housekeeping gene, as PPIB is missing in a large number

```
# 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("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")
```

Genes

```
hm$Mouse_ID <- str_replace(hm$Mouse_ID, "_", "")

field <- hm %>%
  dplyr::filter(origin == "Field")

field <- unique(field)
```

```r
genes_mouse_field <- field %>%
  dplyr::select(c(Mouse_ID, all_of(Genes_wild), GAPDH))

genes_field <- genes_mouse_field  %>%
  dplyr::select(-Mouse_ID)
#remove rows with only nas
genes_field <- genes_field[,colSums(is.na(genes_field))<nrow(genes_field)]
#remove colums with only nas
genes_field <- genes_field[rowSums(is.na(genes_field)) != ncol(genes_field), ]
genes_mouse_field <- genes_mouse_field[row.names(genes_field), ]

##select same rows in the first table
field <- field[row.names(genes_field), ]


###############lab
#select the genes and lab muce
lab <- hm %>%
  dplyr::filter(origin == "Lab", Position == "mLN") #selecting for mln to avoid
# duplicates
lab <- unique(lab)
gene_lab_mouse <- lab %>%
  dplyr::select(c(Mouse_ID, "IFNy", "CXCR3", "IL.6", "IL.13", "IL.10",                 "IL1RN","CASP1",

gene_lab_mouse <- unique(gene_lab_mouse)

genes_lab <- gene_lab_mouse[, -1]

#remove rows with only nas
genes_lab <- genes_lab[,colSums(is.na(genes_lab))<nrow(genes_lab)]

#remove colums with only nas
genes_lab <- genes_lab[rowSums(is.na(genes_lab)) != ncol(genes_lab), ]

genes_lab <- unique(genes_lab)

#select same rows in the first table
gene_lab_mouse <- gene_lab_mouse[row.names(genes_lab), ]

##select same rows in the first table
lab <- lab[row.names(genes_lab), ]
```

Do we have all the measurements of our housekeeping genes?

```r
#glimpse(hm_selection_g)

#dplyr::select(-Mouse_ID)
# looking at patterns of nas)
#pattern_na <-as.data.frame(md.pattern(field_genes))
sapply(field %>%
         dplyr::select(c(all_of(Genes_wild), "PPIB", "GAPDH")),
                       function(x) sum(is.na(x)))
```

```
##   IFNy  CXCR3   IL.6  IL.13  IL.10  IL1RN  CASP1  CXCL9   IDO1  IRGM1    MPO
##     35    110     99     28    217     31    128     42     29     11     39
##   MUC2 MUC5AC  MYD88   NCR1   PRF1 RETNLB  SOCS1 TICAM1    TNF   PPIB  GAPDH
##     14     30     20    126    135    108     11    118     40    212     10
```

```
sapply(lab %>%
         dplyr::select(c("IFNy", "CXCR3", "IL.6", "IL.13", "IL.10",                 "IL1RN","CASP1", "C
                      function(x) sum(is.na(x)))
```

```
##   IFNy  CXCR3   IL.6  IL.13  IL.10  IL1RN  CASP1  CXCL9   IDO1  IRGM1    MPO
##     27      0     12     96     13      0      3      0      0      0     15
##   MUC2 MUC5AC  MYD88   NCR1   PRF1 RETNLB  SOCS1 TICAM1    TNF   PPIB  GAPDH
##      0      0      0     13     23      0      0      3      2      1     96
```

Gene normalization

For the field samples the missing values in the house-keeping genes are: PPIB: 212 GAPDH: 10

For the lab samples the missing values in the house-keeping genes are: PPIB: 1 GAPDH: 96

Downlad and install the package for Gene normalization

Step 1. Normalize to (REF): DCq = Cq (TAR) – Cq (REF) Step 2. Exponential expression transform: DCq Expression = 2–DCq Step 3. Average replicates and calculate standard deviation Step 4. Normalize to treatment control Step 5. % KD = (1 – DDCq) × 100

```
####################### field ##########################
# select first the field samples

df <- genes_mouse_field

############### IL.13
# dct
df <- df %>%
  mutate(IL.13_dct = IL.13 - GAPDH)
# mean of dct
dct_mean <- mean(df$IL.13_dct, na.rm = TRUE)
#fold gene expression
 df <- df %>%
   mutate(IL.13_N = 2^ - (IL.13_dct - dct_mean)) %>%
   mutate(IL.13_N = round(IL.13_N, digits = 2))

############# I tried writing a nice function but I failed so now I am
 # going t repeat this many many times

 ############### IFNy
 # dct
df <- df %>%
  mutate(IFNy_dct = IFNy - GAPDH)

# mean of dct
dct_mean <- mean(df$IFNy_dct, na.rm = TRUE)

#fold gene expression
 df <- df %>%
   mutate(IFNy_N = 2^ - (IFNy_dct - dct_mean)) %>%
   mutate(IFNy_N = round(IFNy_N, digits = 2))
 ############### CXCR3
 df <- df %>%
  mutate(CXCR3_dct = CXCR3 - GAPDH)

# mean of dct
dct_mean <- mean(df$CXCR3_dct, na.rm = TRUE)

#fold gene expression
 df <- df %>%
   mutate(CXCR3_N = 2^ - (CXCR3_dct - dct_mean)) %>%
   mutate(CXCR3_N = round(CXCR3_N, digits = 2))

 ############### IL.6
 df <- df %>%
  mutate(IL.6_dct = IL.6 - GAPDH)

# mean of dct
dct_mean <- mean(df$IL.6_dct, na.rm = TRUE)

#fold gene expression
 df <- df %>%
   mutate(IL.6_N = 2^ - (IL.6_dct - dct_mean)) %>%
   mutate(IL.6_N = round(IL.6_N, digits = 2))

 ##############  IL1RN
 df <- df %>%
  mutate(IL1RN_dct = IL1RN - GAPDH)

# mean of dct
dct_mean <- mean(df$IL1RN_dct, na.rm = TRUE)

#fold gene expression
 df <- df %>%
   mutate(IL1RN_N = 2^ - (IL1RN_dct - dct_mean)) %>%
   mutate(IL1RN_N = round(IL1RN_N, digits = 2))


 ##############  CASP1
 df <- df %>%
  mutate(CASP1_dct = CASP1 - GAPDH)

# mean of dct
dct_mean <- mean(df$CASP1_dct, na.rm = TRUE)

#fold gene expression
 df <- df %>%
   mutate(CASP1_N = 2^ - (CASP1_dct - dct_mean)) %>%
   mutate(CASP1_N = round(CASP1_N, digits = 2))


 ##############  CXCL9
 df <- df %>%
  mutate(CXCL9_dct = CXCL9 - GAPDH)

# mean of dct
dct_mean <- mean(df$CXCL9_dct, na.rm = TRUE)

#fold gene expression
 df <- df %>%
   mutate(CXCL9_N = 2^ - (CXCL9_dct - dct_mean)) %>%
   mutate(CXCL9_N = round(CXCL9_N, digits = 2))

  ##############  IDO1
 df <- df %>%
  mutate(IDO1_dct = IDO1 - GAPDH)
# mean of dct
dct_mean <- mean(df$IDO1_dct, na.rm = TRUE)
#fold gene expression
 df <- df %>%
   mutate(IDO1_N = 2^ - (IDO1_dct - dct_mean)) %>%
   mutate(IDO1_N = round(IDO1_N, digits = 2))

  ##############  IRGM1
 df <- df %>%
  mutate(IRGM1_dct = IRGM1 - GAPDH)
# mean of dct
dct_mean <- mean(df$IRGM1_dct, na.rm = TRUE)
#fold gene expression
 df <- df %>%
   mutate(IRGM1_N = 2^ - (IRGM1_dct - dct_mean)) %>%
   mutate(IRGM1_N = round(IRGM1_N, digits = 2))

  ##############  MPO
 df <- df %>%
  mutate(MPO_dct = MPO - GAPDH)
# mean of dct
dct_mean <- mean(df$MPO_dct, na.rm = TRUE)
#fold gene expression
 df <- df %>%
   mutate(MPO_N = 2^ - (MPO_dct - dct_mean)) %>%
   mutate(MPO_N = round(MPO_N, digits = 2))


  ##############  MUC2
 df <- df %>%
  mutate(MUC2_dct = MUC2 - GAPDH)
# mean of dct
dct_mean <- mean(df$MUC2_dct, na.rm = TRUE)
#fold gene expression
 df <- df %>%
   mutate(MUC2_N = 2^ - (MUC2_dct - dct_mean)) %>%
   mutate(MUC2_N = round(MUC2_N, digits = 2))

  ##############  MUC5AC
 df <- df %>%
  mutate(MUC5AC_dct = MUC5AC - GAPDH)
# mean of dct
dct_mean <- mean(df$MUC5AC_dct, na.rm = TRUE)
#fold gene expression
 df <- df %>%
   mutate(MUC5AC_N = 2^ - (MUC5AC_dct - dct_mean)) %>%
   mutate(MUC5AC_N = round(MUC5AC_N, digits = 2))

  ##############  MYD88
 df <- df %>%
  mutate(MYD88_dct = MYD88 - GAPDH)
# mean of dct
dct_mean <- mean(df$MYD88_dct, na.rm = TRUE)
#fold gene expression
 df <- df %>%
   mutate(MYD88_N = 2^ - (MYD88_dct - dct_mean)) %>%
   mutate(MYD88_N = round(MYD88_N, digits = 2))

  ##############  NCR1
 df <- df %>%
  mutate(NCR1_dct = NCR1 - GAPDH)
# mean of dct
dct_mean <- mean(df$NCR1_dct, na.rm = TRUE)
#fold gene expression
 df <- df %>%
   mutate(NCR1_N = 2^ - (NCR1_dct - dct_mean)) %>%
   mutate(NCR1_N = round(NCR1_N, digits = 2))


  ##############  PRF1
 df <- df %>%
  mutate(PRF1_dct = PRF1 - GAPDH)
# mean of dct
dct_mean <- mean(df$PRF1_dct, na.rm = TRUE)
#fold gene expression
 df <- df %>%
   mutate(PRF1_N = 2^ - (PRF1_dct - dct_mean)) %>%
   mutate(PRF1_N = round(PRF1_N, digits = 2))

  ##############  RETNLB
 df <- df %>%
  mutate(RETNLB_dct = RETNLB - GAPDH)
# mean of dct
dct_mean <- mean(df$RETNLB_dct, na.rm = TRUE)
#fold gene expression
 df <- df %>%
   mutate(RETNLB_N = 2^ - (RETNLB_dct - dct_mean)) %>%
   mutate(RETNLB_N = round(RETNLB_N, digits = 2))



  ##############  SOCS1
 df <- df %>%
  mutate(SOCS1_dct = SOCS1 - GAPDH)
# mean of dct
dct_mean <- mean(df$SOCS1_dct, na.rm = TRUE)
#fold gene expression
 df <- df %>%
   mutate(SOCS1_N = 2^ - (SOCS1_dct - dct_mean)) %>%
   mutate(SOCS1_N = round(SOCS1_N, digits = 2))

  ##############  TICAM1
 df <- df %>%
  mutate(TICAM1_dct = TICAM1 - GAPDH)
# mean of dct
dct_mean <- mean(df$TICAM1_dct, na.rm = TRUE)
#fold gene expression
 df <- df %>%
   mutate(TICAM1_N = 2^ - (TICAM1_dct - dct_mean)) %>%
   mutate(TICAM1_N = round(TICAM1_N, digits = 2))

 ##############  TNF
 df <- df %>%
  mutate(TNF_dct = TNF - GAPDH)
# mean of dct
dct_mean <- mean(df$TNF_dct, na.rm = TRUE)
#fold gene expression
 df <- df %>%
   mutate(TNF_N = 2^ - (TNF_dct - dct_mean)) %>%
   mutate(TNF_N = round(TNF_N, digits = 2))


 df -> df_field
```

Genes Lab

```
 ################################# lab
 # select first the field samples


df_lab <- gene_lab_mouse

  ############### IFNy
  df_lab <- df_lab %>%
  mutate(IFNy_dct = IFNy - PPIB)
# mean of dct
dct_mean <- mean(df_lab$IFNy_dct, na.rm = TRUE)
#fold gene expression
 df_lab <- df_lab %>%
   mutate(IFNy_N = 2^ - (IFNy_dct - dct_mean)) %>%
   mutate(IFNy_N = round(IFNy_N, digits = 2))



 ############### CXCR3
  df_lab <- df_lab %>%
  mutate(CXCR3_dct = CXCR3 - PPIB)
```

```
# mean of dct
dct_mean <- mean(df_lab$CXCR3_dct, na.rm = TRUE)
#fold gene expression
 df_lab <- df_lab %>%
   mutate(CXCR3_N = 2^ - (CXCR3_dct - dct_mean)) %>%
   mutate(CXCR3_N = round(CXCR3_N, digits = 2))

 ############### IL.6
   df_lab <- df_lab %>%
  mutate(IL.6_dct = IL.6 - PPIB)
# mean of dct
dct_mean <- mean(df_lab$IL.6_dct, na.rm = TRUE)
#fold gene expression
 df_lab <- df_lab %>%
   mutate(IL.6_N = 2^ - (IL.6_dct - dct_mean)) %>%
   mutate(IL.6_N = round(IL.6_N, digits = 2))


 ############## IL.13
   df_lab <- df_lab %>%
  mutate(IL.13_dct = IL.13 - PPIB)
# mean of dct
dct_mean <- mean(df_lab$IL.13_dct, na.rm = TRUE)
#fold gene expression
 df_lab <- df_lab %>%
   mutate(IL.13_N = 2^ - (IL.13_dct - dct_mean)) %>%
   mutate(IL.13_N = round(IL.13_N, digits = 2))

 ##############  IL1RN
   df_lab <- df_lab %>%
  mutate(IL1RN_dct = IL1RN - PPIB)
# mean of dct
dct_mean <- mean(df_lab$IL1RN_dct, na.rm = TRUE)
#fold gene expression
 df_lab <- df_lab %>%
   mutate(IL1RN_N = 2^ - (IL1RN_dct - dct_mean)) %>%
   mutate(IL1RN_N = round(IL1RN_N, digits = 2))

 ##############  CASP1
   df_lab <- df_lab %>%
  mutate(CASP1_dct = CASP1 - PPIB)
# mean of dct
dct_mean <- mean(df_lab$CASP1_dct, na.rm = TRUE)
#fold gene expression
 df_lab <- df_lab %>%
   mutate(CASP1_N = 2^ - (CASP1_dct - dct_mean)) %>%
   mutate(CASP1_N = round(CASP1_N, digits = 2))


 ##############  CXCL9
   df_lab <- df_lab %>%
  mutate(CXCL9_dct = CXCL9 - PPIB)
# mean of dct
```

```r
dct_mean <- mean(df_lab$CXCL9_dct, na.rm = TRUE)
#fold gene expression
 df_lab <- df_lab %>%
   mutate(CXCL9_N = 2^ - (CXCL9_dct - dct_mean)) %>%
   mutate(CXCL9_N = round(CXCL9_N, digits = 2))


  ##############  IDO1
   df_lab <- df_lab %>%
  mutate(IDO1_dct = IDO1 - PPIB)
# mean of dct
dct_mean <- mean(df_lab$IDO1_dct, na.rm = TRUE)
#fold gene expression
 df_lab <- df_lab %>%
   mutate(IDO1_N = 2^ - (IDO1_dct - dct_mean)) %>%
   mutate(IDO1_N = round(IDO1_N, digits = 2))

  ##############  IRGM1
   df_lab <- df_lab %>%
  mutate(IRGM1_dct = IRGM1 - PPIB)
# mean of dct
dct_mean <- mean(df_lab$IRGM1_dct, na.rm = TRUE)
#fold gene expression
 df_lab <- df_lab %>%
   mutate(IRGM1_N = 2^ - (IRGM1_dct - dct_mean)) %>%
   mutate(IRGM1_N = round(IRGM1_N, digits = 2))

  ##############  MPO
   df_lab <- df_lab %>%
  mutate(MPO_dct = MPO - PPIB)
# mean of dct
dct_mean <- mean(df_lab$MPO_dct, na.rm = TRUE)
#fold gene expression
 df_lab <- df_lab %>%
   mutate(MPO_N = 2^ - (MPO_dct - dct_mean)) %>%
   mutate(MPO_N = round(MPO_N, digits = 2))

  ##############  MUC2
   df_lab <- df_lab %>%
  mutate(MUC2_dct = MUC2 - PPIB)
# mean of dct
dct_mean <- mean(df_lab$MUC2_dct, na.rm = TRUE)
#fold gene expression
 df_lab <- df_lab %>%
   mutate(MUC2_N = 2^ - (MUC2_dct - dct_mean)) %>%
   mutate(MUC2_N = round(MUC2_N, digits = 2))

  ##############  MUC5AC
   df_lab <- df_lab %>%
  mutate(MUC5AC_dct = MUC5AC - PPIB)
# mean of dct
dct_mean <- mean(df_lab$MUC5AC_dct, na.rm = TRUE)
#fold gene expression
```

```
 df_lab <- df_lab %>%
   mutate(MUC5AC_N = 2^ - (MUC5AC_dct - dct_mean)) %>%
   mutate(MUC5AC_N = round(MUC5AC_N, digits = 2))


  ##############  MYD88
   df_lab <- df_lab %>%
  mutate(MYD88_dct = MYD88 - PPIB)
# mean of dct
dct_mean <- mean(df_lab$MYD88_dct, na.rm = TRUE)
#fold gene expression
 df_lab <- df_lab %>%
   mutate(MYD88_N = 2^ - (MYD88_dct - dct_mean)) %>%
   mutate(MYD88_N = round(MYD88_N, digits = 2))


  ##############  NCR1
   df_lab <- df_lab %>%
  mutate(NCR1_dct = NCR1 - PPIB)
# mean of dct
dct_mean <- mean(df_lab$NCR1_dct, na.rm = TRUE)
#fold gene expression
 df_lab <- df_lab %>%
   mutate(NCR1_N = 2^ - (NCR1_dct - dct_mean)) %>%
   mutate(NCR1_N = round(NCR1_N, digits = 2))

  ##############  PRF1
   df_lab <- df_lab %>%
  mutate(PRF1_dct = PRF1 - PPIB)
# mean of dct
dct_mean <- mean(df_lab$PRF1_dct, na.rm = TRUE)
#fold gene expression
 df_lab <- df_lab %>%
   mutate(PRF1_N = 2^ - (PRF1_dct - dct_mean)) %>%
   mutate(PRF1_N = round(PRF1_N, digits = 2))

  ##############  RETNLB
   df_lab <- df_lab %>%
  mutate(RETNLB_dct = RETNLB - PPIB)
# mean of dct
dct_mean <- mean(df_lab$RETNLB_dct, na.rm = TRUE)
#fold gene expression
 df_lab <- df_lab %>%
   mutate(RETNLB_N = 2^ - (RETNLB_dct - dct_mean)) %>%
   mutate(RETNLB_N = round(RETNLB_N, digits = 2))

  ##############  SOCS1
   df_lab <- df_lab %>%
  mutate(SOCS1_dct = SOCS1 - PPIB)
# mean of dct
dct_mean <- mean(df_lab$SOCS1_dct, na.rm = TRUE)
#fold gene expression
 df_lab <- df_lab %>%
   mutate(SOCS1_N = 2^ - (SOCS1_dct - dct_mean)) %>%
   mutate(SOCS1_N = round(SOCS1_N, digits = 2))

  ##############  TICAM1
   df_lab <- df_lab %>%
  mutate(TICAM1_dct = TICAM1 - PPIB)
# mean of dct
dct_mean <- mean(df_lab$TICAM1_dct, na.rm = TRUE)
#fold gene expression
 df_lab <- df_lab %>%
   mutate(TICAM1_N = 2^ - (TICAM1_dct - dct_mean)) %>%
   mutate(TICAM1_N = round(TICAM1_N, digits = 2))

 ##############  TNF
   df_lab <- df_lab %>%
  mutate(TNF_dct = TNF - PPIB)
# mean of dct
dct_mean <- mean(df_lab$TNF_dct, na.rm = TRUE)
#fold gene expression
 df_lab <- df_lab %>%
   mutate(TNF_N = 2^ - (TNF_dct - dct_mean)) %>%
   mutate(TNF_N = round(TNF_N, digits = 2))
```

#now join the data again

without the housekeeping genes

First clean and remove the non normalizes genes

```
#df_lab <- df_lab %>%
#  dplyr::select(-c("IFNy", "CXCR3", "IL.6", "IL.13", "IL.10",
#"IL1RN","CASP1", "CXCL9", "IDO1", "IRGM1", "MPO",   "MUC2", "MUC5AC", "MYD88",
#"NCR1", "PRF1", "RETNLB", "SOCS1",   "TICAM1", "TNF", PPIB, contains("_dct")))

# remove ending _N
#df_lab <- df_lab %>%
#  rename_with(~str_remove(.x, "_N"))

#df_field <- df_field %>%
#  dplyr::select(-c(all_of(Genes_wild), GAPDH, contains("_dct")))

# remove ending _N
#df_field <- df_field %>%
 # rename_with(~str_remove(.x, "_N"))




###################### let's try just using DCT
df_lab <- df_lab %>%
 dplyr::select(-c("IFNy", "CXCR3", "IL.6", "IL.13", "IL.10", "IL1RN","CASP1",
                  "CXCL9", "IDO1", "IRGM1", "MPO", "MUC2", "MUC5AC", "MYD88",
                  "NCR1", "PRF1", "RETNLB", "SOCS1",  "TICAM1", "TNF", PPIB,
                  contains("_N")))
```

```
# remove ending _dct
df_lab <- df_lab %>%
  rename_with(~str_remove(.x, "_dct"))

df_field <- df_field %>%
  dplyr::select(-c("IFNy", "CXCR3", "IL.6", "IL.13", "IL.10", "IL1RN","CASP1",
                   "CXCL9", "IDO1", "IRGM1", "MPO", "MUC2", "MUC5AC", "MYD88",
                   "NCR1", "PRF1", "RETNLB", "SOCS1",  "TICAM1", "TNF", GAPDH,
                   contains("_N")))

# remove ending
df_field <- df_field %>%
  rename_with(~str_remove(.x, "_dct"))


# add the new genes to the complete data sets
lab <- lab %>%
 dplyr::select(-c("IFNy", "CXCR3", "IL.6", "IL.13", "IL.10", "IL1RN","CASP1",
                "CXCL9", "IDO1", "IRGM1", "MPO",  "MUC2", "MUC5AC", "MYD88",
                "NCR1", "PRF1", "RETNLB", "SOCS1",  "TICAM1", "TNF")) %>%
  left_join(df_lab, by = "Mouse_ID")

field <- field %>%
 dplyr::select(-c("IFNy", "CXCR3", "IL.6", "IL.13", "IL.10", "IL1RN","CASP1",
                "CXCL9", "IDO1", "IRGM1", "MPO",  "MUC2", "MUC5AC", "MYD88",
                "NCR1", "PRF1", "RETNLB", "SOCS1",  "TICAM1", "TNF")) %>%
  left_join(df_field, by = "Mouse_ID")


hm_norm <- rbind(lab, field)
```

```
 ##save the imputed data
write.csv(hm_norm, "output_data/2.1.norm_MICE_data_set.csv", row.names = FALSE)
```