Common Marmoset Gut Microbiome Profiles in Health and Intestinal Disease

Alex Sheh

September 21, 2020

This is the 2nd part of the RNAseq analysis following FASTQ processing and alignment. This section focuses on duodenal stricture samples

We load rdata file with the feature counts and libraries.

```
load("duo_cj_feature_counts.RData")
load("duo_hs_feature_counts.RData")

# for ML algorithms
library(Rsubread)
library(edgeR)
```

```
## Loading required package: limma
```

```
library(gplots)
```

```
##
## Attaching package: 'gplots'

## The following object is masked from 'package:stats':
##
##     lowess
```

```
library(org.Hs.eg.db)
```

```
## Loading required package: AnnotationDbi

## Loading required package: stats4

## Loading required package: BiocGenerics

## Loading required package: parallel
```

```
##
## Attaching package: 'BiocGenerics'

## The following objects are masked from 'package:parallel':
##
##     clusterApply, clusterApplyLB, clusterCall, clusterEvalQ,
##     clusterExport, clusterMap, parApply, parCapply, parLapply,
##     parLapplyLB, parRapply, parSapply, parSapplyLB

## The following object is masked from 'package:limma':
##
##     plotMA

## 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, which.max, which.min

## 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")'.

## Loading required package: IRanges

## Loading required package: S4Vectors

##
## Attaching package: 'S4Vectors'

## The following object is masked from 'package:gplots':
##
##     space

## The following object is masked from 'package:base':
##
##     expand.grid

##
## Attaching package: 'IRanges'
```

```
## The following object is masked from 'package:grDevices':
##
##     windows
```

```
##
```

```
library(AnnotationDbi)
library(GO.db)
```

```
##
```

```
library(mygene)
```

```
## Loading required package: GenomicFeatures
```

```
## Loading required package: GenomeInfoDb
```

```
## Loading required package: GenomicRanges
```

```
library(topGO)
```

```
## Loading required package: graph
```

```
## Loading required package: SparseM
```

```
##
## Attaching package: 'SparseM'
```

```
## The following object is masked from 'package:base':
##
##     backsolve
```

```
##
## groupGOTerms:    GOBPTerm, GOMFTerm, GOCCTerm environments built.
```

```
##
## Attaching package: 'topGO'
```

```
## The following object is masked from 'package:GenomicFeatures':
##
##     genes
```

```
## The following object is masked from 'package:IRanges':
##
##     members
```

```
data("geneList")
library(Rgraphviz)
```

```
## Loading required package: grid
```

```
##
## Attaching package: 'grid'
```

```
## The following object is masked from 'package:topGO':
##
##     depth
```

```
##
## Attaching package: 'Rgraphviz'
```

```
## The following objects are masked from 'package:IRanges':
##
##     from, to
```

```
## The following objects are masked from 'package:S4Vectors':
##
##     from, to
```

```
library(ggplot2)
library(colorspace)
library(ggVennDiagram)

sessionInfo()
```

```
## R version 3.6.3 (2020-02-29)
## Platform: x86_64-w64-mingw32/x64 (64-bit)
## Running under: Windows 10 x64 (build 18363)
##
## Matrix products: default
##
## locale:
## [1] LC_COLLATE=English_United States.1252
## [2] LC_CTYPE=English_United States.1252
## [3] LC_MONETARY=English_United States.1252
## [4] LC_NUMERIC=C
## [5] LC_TIME=English_United States.1252
##
## attached base packages:
##  [1] grid      parallel  stats4    stats     graphics  grDevices utils
##  [8] datasets  methods   base
##
## other attached packages:
##  [1] ggVennDiagram_0.3      colorspace_1.4-1       ggplot2_3.3.2
##  [4] Rgraphviz_2.30.0       topGO_2.38.1           SparseM_1.78
##  [7] graph_1.64.0           mygene_1.22.0          GenomicFeatures_1.38.2
## [10] GenomicRanges_1.38.0   GenomeInfoDb_1.22.1    GO.db_3.10.0
```

```
## [13] org.Hs.eg.db_3.10.0    AnnotationDbi_1.48.0   IRanges_2.20.2
## [16] S4Vectors_0.24.4       Biobase_2.46.0         BiocGenerics_0.32.0
## [19] gplots_3.0.4           edgeR_3.28.1           limma_3.42.2
## [22] Rsubread_2.0.1
##
## loaded via a namespace (and not attached):
##   [1] class_7.3-15                ellipsis_0.3.1
##   [3] htmlTable_2.0.1             futile.logger_1.4.3
##   [5] XVector_0.26.0              base64enc_0.1-3
##   [7] rstudioapi_0.11             bit64_0.9-7
##   [9] sqldf_0.4-11                splines_3.6.3
##  [11] knitr_1.29                  Formula_1.2-3
##  [13] jsonlite_1.7.0              Rsamtools_2.2.3
##  [15] cluster_2.1.0               dbplyr_1.4.4
##  [17] png_0.1-7                   compiler_3.6.3
##  [19] httr_1.4.2                  backports_1.1.7
##  [21] assertthat_0.2.1            Matrix_1.2-18
##  [23] formatR_1.7                 acepack_1.4.1
##  [25] htmltools_0.5.0             prettyunits_1.1.1
##  [27] tools_3.6.3                 gtable_0.3.0
##  [29] glue_1.4.1                  GenomeInfoDbData_1.2.2
##  [31] dplyr_1.0.0                 rappdirs_0.3.1
##  [33] Rcpp_1.0.5                  vctrs_0.3.1
##  [35] Biostrings_2.54.0           gdata_2.18.0
##  [37] rtracklayer_1.46.0          xfun_0.16
##  [39] stringr_1.4.0               proto_1.0.0
##  [41] lifecycle_0.2.0             gtools_3.8.2
##  [43] XML_3.99-0.3                zlibbioc_1.32.0
##  [45] scales_1.1.1                hms_0.5.3
##  [47] SummarizedExperiment_1.16.1 lambda.r_1.2.4
##  [49] RColorBrewer_1.1-2          yaml_2.2.1
##  [51] curl_4.3                    memoise_1.1.0
##  [53] gridExtra_2.3               biomaRt_2.42.1
##  [55] rpart_4.1-15                latticeExtra_0.6-29
##  [57] stringi_1.4.6               RSQLite_2.2.0
##  [59] e1071_1.7-3                 checkmate_2.0.0
##  [61] caTools_1.18.0              BiocParallel_1.20.1
##  [63] chron_2.3-55                rlang_0.4.7
##  [65] pkgconfig_2.0.3             matrixStats_0.56.0
##  [67] bitops_1.0-6                evaluate_0.14
##  [69] lattice_0.20-38             sf_0.9-5
##  [71] purrr_0.3.4                 GenomicAlignments_1.22.1
##  [73] htmlwidgets_1.5.1           bit_1.1-15.2
##  [75] tidyselect_1.1.0            plyr_1.8.6
##  [77] magrittr_1.5                R6_2.4.1
##  [79] generics_0.0.2              Hmisc_4.4-0
##  [81] DelayedArray_0.12.3         DBI_1.1.0
##  [83] withr_2.2.0                 gsubfn_0.7
##  [85] pillar_1.4.6                foreign_0.8-75
##  [87] units_0.6-7                 survival_3.1-8
##  [89] RCurl_1.98-1.2              nnet_7.3-12
##  [91] tibble_3.0.3                crayon_1.3.4
##  [93] futile.options_1.0.1        KernSmooth_2.23-16
##  [95] BiocFileCache_1.10.2        rmarkdown_2.3
```

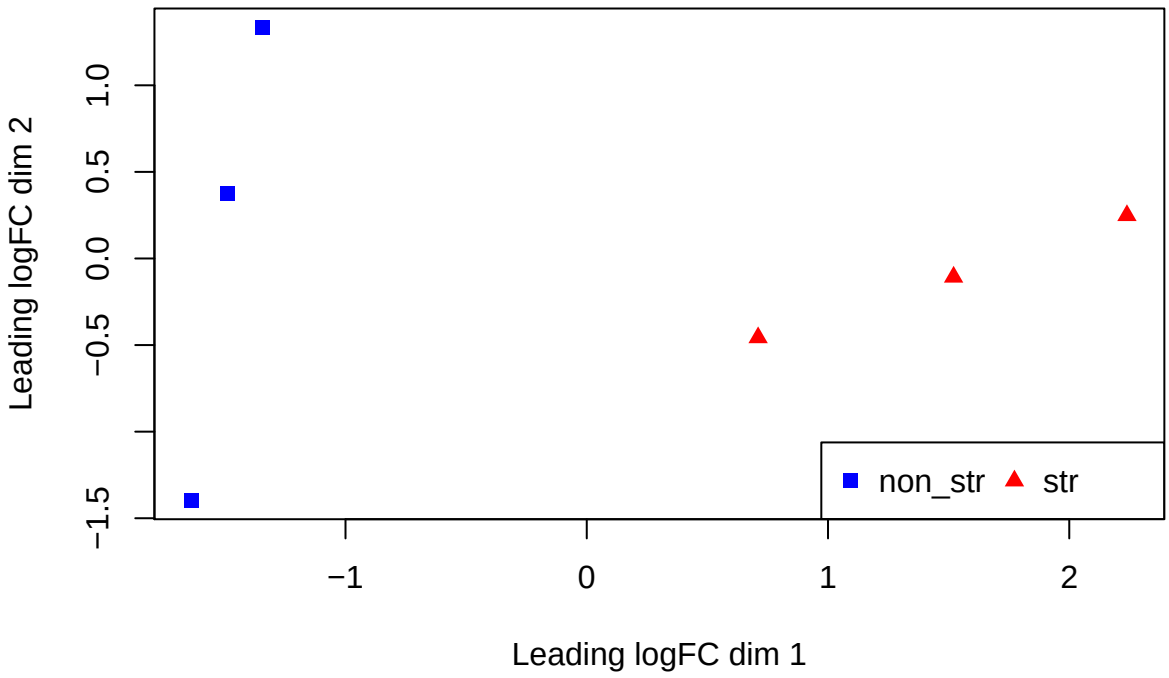
```
##  [97] jpeg_0.1-8.1                progress_1.2.2
##  [99] locfit_1.5-9.4              data.table_1.12.8
## [101] blob_1.2.1                  classInt_0.4-3
## [103] digest_0.6.25               VennDiagram_1.6.20
## [105] openssl_1.4.2               munsell_0.5.0
## [107] askpass_1.1
```

```
##########function getgeneid
getgeneid<-function(strGO,isde){
  require(org.Hs.eg.db)
  require(mygene)
  require(AnnotationDbi)
  x <- org.Hs.egGO2ALLEGS
  Rkeys(x) <- strGO
  EG <- mappedLkeys(x)

  is.deh.keep<-isde@.Data!=0
  is.deh.kept <- as.matrix(isde[is.deh.keep])
  rnames <-rownames(isde)
  rnames <- rnames[is.deh.keep]
  rownames(is.deh.kept)<-rnames
  isde_GO <- intersect(EG, rownames(is.deh.kept))
  GOI<-as.matrix(is.deh.kept[isde_GO,])
  return(getGenes(rownames(GOI), fields = c("symbol","name","summary")))
}
```

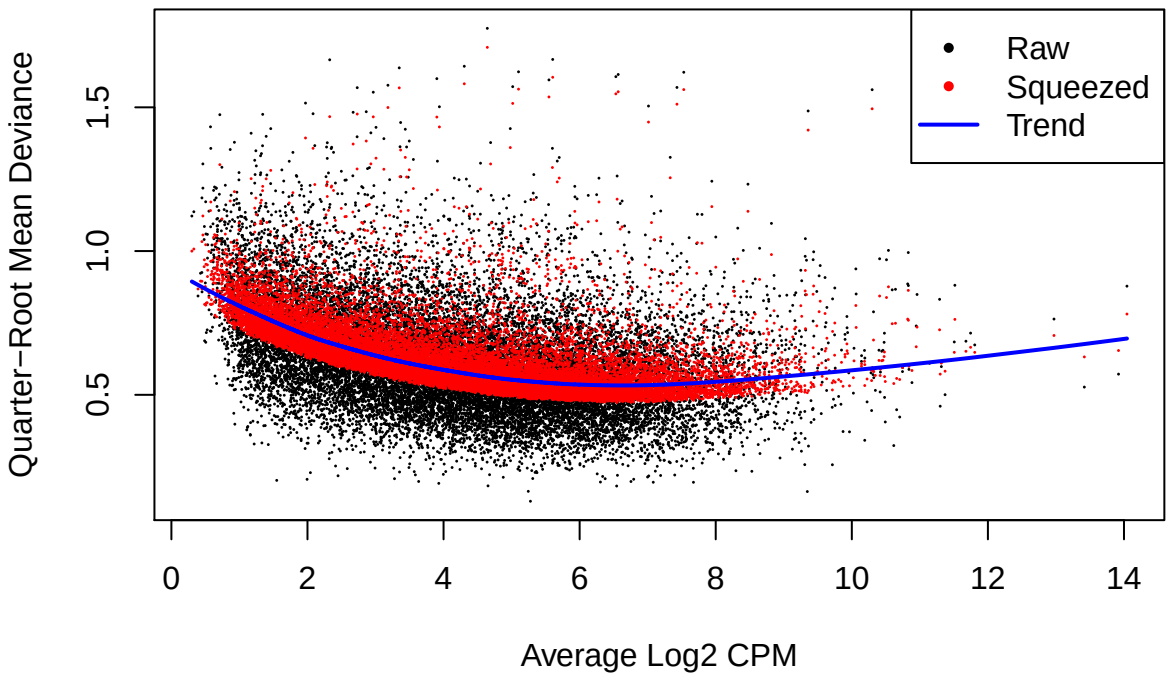
Compare stricture cases

```
#tabulate
Duo_6 <- cbind(fc_3$counts, fc_7$counts, fc_9$counts, fc_5$counts,
               fc_14$counts, fc_21$counts)
# write.table(Duo_6, file="20200824_raw_feature_counts_duo_3v3p.txt")

g_3v3p <- c("duo_non","duo_non","duo_non","duo_str","duo_str","duo_str")

#trim lowly expressed exons
Duo_3v3p <- DGEList(counts = Duo_6[,1:6], group = g_3v3p)
# dim(Duo_3v3p)

#counts per million have to have counts greater than X per million
# and be found in at least Y samples
round_3v3p <- as.numeric(round(cpm(10, mean(Duo_3v3p$samples$lib.size))))
#this value is used in the rowsum inequality
keep_3v3p <-rowSums(cpm(Duo_3v3p)>round_3v3p) >= 2
Duo_3v3p <- Duo_3v3p[keep_3v3p,]
# dim(Duo_3v3p)

#TMM normalization
Duo_3v3p.norm <-calcNormFactors(Duo_3v3p, method = "TMM")
Duo_3v3p.norm$samples
```

```
# #plot MA plot - mean and difference plot based on the first sample.
# plotMD(cpm(Duo_3v3p.norm, log=TRUE), column=1)
# abline(h=0, col="red", lty=2, lwd=2)

#data exploration of dataset
plotMDS(Duo_3v3p.norm,pch = c(15,15,15,17,17,17),
        col = c("blue", "blue","blue", "red","red","red"))
legend("bottomright", legend=c("non_str", "str"), pch=c(15,17),
       col=c("blue","red"), ncol=2)
```

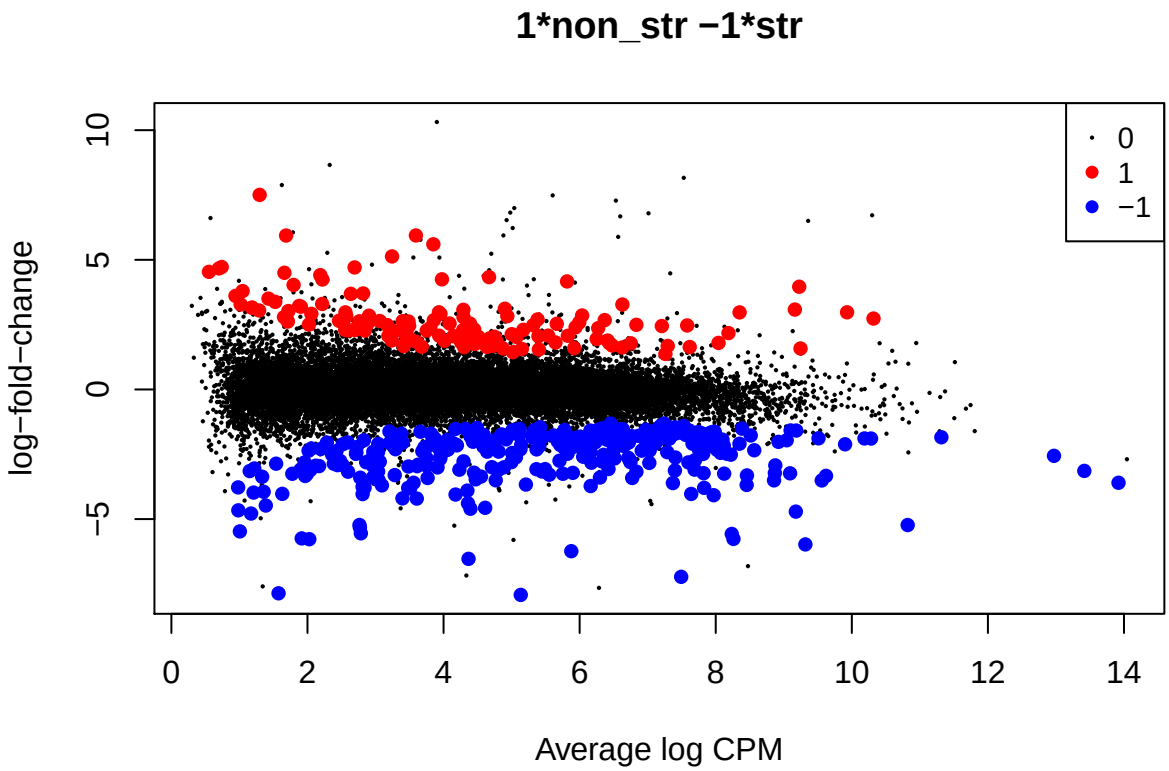



```
# dev.copy(png, paste0("str_3v3p_plotMDS.png"))
# dev.off()

#Estimate common dispersion (could just return it to Duo_str.norm as it just adds columns)
des_3v3p <- model.matrix(~ 0 + g_3v3p)
colnames(des_3v3p) <- c("non_str","str")

Duo_3v3p.norm <- estimateCommonDisp(Duo_3v3p.norm, des_3v3p, verbose = TRUE, robust= TRUE)
# write.table(Duo_3v3p.norm$counts, file="20200824_duo_3v3p_norm_counts.txt")

fit_3v3p <- glmQLFit(Duo_3v3p.norm, des_3v3p, robust=TRUE)
head(fit_3v3p$coefficients)
plotQLDisp(fit_3v3p)
```




```
# dev.copy(png, paste0("str_3v3p_plotQLDisp.png"))
# dev.off()
summary(fit_3v3p$df.prior)

# outliers from the mean-NB dispersion trend. Outliers are marked by small prior.df values:
o_3v3p <- order(fit_3v3p$df.prior)
Duo_3v3p.norm$counts[o_3v3p[1:6],] #genes tag used in the tutorial but our labels had names

con_3v3p <- makeContrasts(non_str - str, levels=des_3v3p)
res_3v3p <- glmQLFTest(fit_3v3p, contrast=con_3v3p)
topTags(res_3v3p)
#highly expressed in stricture than non-stricture

de_3v3p <- decideTestsDGE(res_3v3p, p.value=0.05)
# summary(de_3v3p)
# plotSmear(res_3v3p, de.tags=rownames(res_3v3p)[de_3v3p!=0])
# dev.copy(png, paste0("str_3v3p_smear.png"))
# dev.off()

#supplementary table 5
write.table(topTags(res_3v3p, n=100000)$table, file="supplementary_table_5.txt")

# #imposing the 1.5 fold change
res.fc_3v3p <- glmTreat(fit_3v3p, contrast = con_3v3p, lfc = log2(1.5))
topTags(res.fc_3v3p)
de.fc_3v3p <- decideTestsDGE(res.fc_3v3p)
```

```
summary(de.fc_3v3p)
plotMD(res.fc_3v3p, status=de.fc_3v3p, values=c(1,-1), col=c("red","blue"), legend="topright")
```

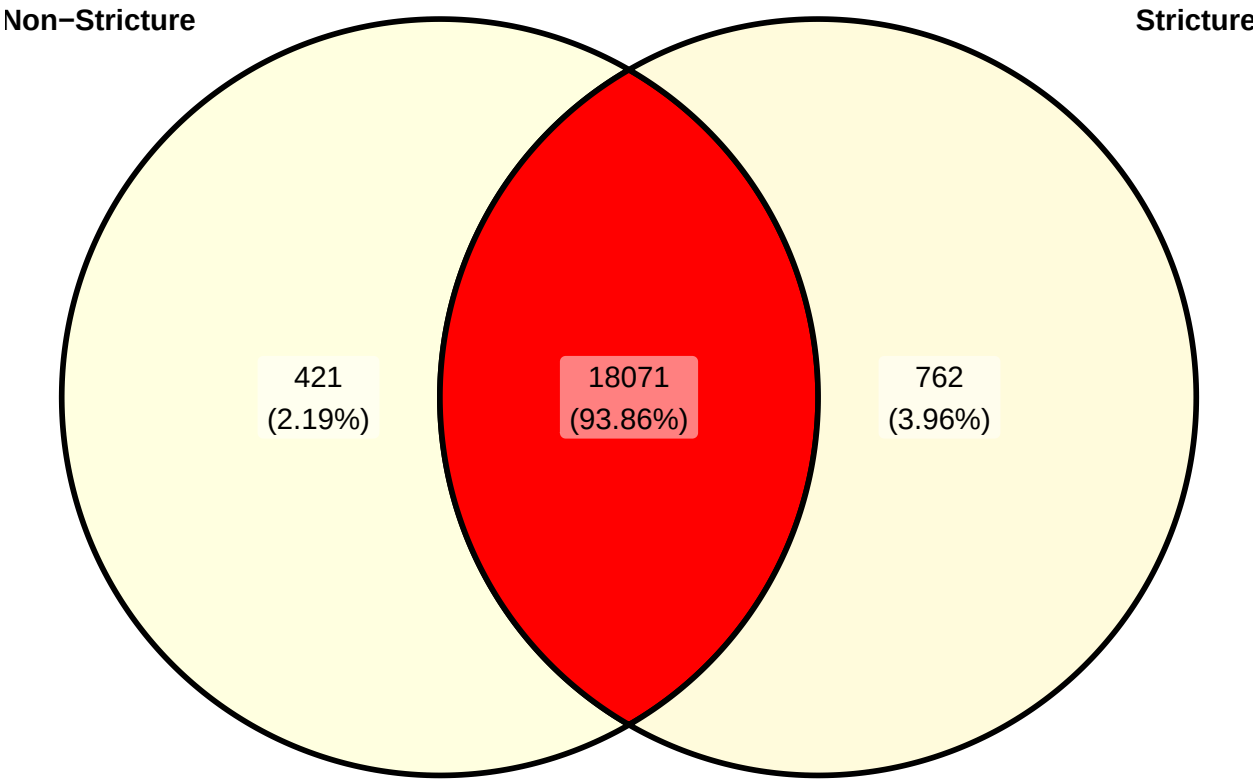



```
# dev.copy(png, paste0("str_3v3p_plotMD.png"))
# dev.off()


# 4b venn str ------------------------------------------------------------------
str_toptags_3v3p <- topTags(res_3v3p, n =2000, p.value = 0.05)
str_toptags_3v3p <- as.data.frame(str_toptags_3v3p[,c(1,5)])
str_toptags_3v3p <- str_toptags_3v3p[order(str_toptags_3v3p$logFC),]
str_toptags_genes_3v3p <- rownames(str_toptags_3v3p)
str_genes_3v3p <- rownames(Duo_3v3p.norm$pseudo.counts)
str_commongenes_3v3p <- setdiff(str_genes_3v3p,str_toptags_genes_3v3p)

str_toptags_3v3p_up <- str_toptags_3v3p$logFC>0
str_toptags_3v3p_up <- str_toptags_3v3p[str_toptags_3v3p_up,]
str_toptags_3v3p_dn <- str_toptags_3v3p$logFC<0
str_toptags_3v3p_dn <- str_toptags_3v3p[str_toptags_3v3p_dn,]
str_genes_3v3p_up <-rownames(str_toptags_3v3p_up)
str_genes_3v3p_dn <-rownames(str_toptags_3v3p_dn)
str_genes_3v3p_com_up <-append(str_genes_3v3p_up,str_commongenes_3v3p)
str_genes_3v3p_com_dn <-append(str_genes_3v3p_dn,str_commongenes_3v3p)
```

```
# draw venn
str_3v3p_venn <- list(Nonstricture=str_genes_3v3p_com_up, Stricture=str_genes_3v3p_com_dn)
str_3v3p_venn_plot <- ggVennDiagram(str_3v3p_venn,color="black",size=1,
                                    category.names = c("         Non-Stricture", "Stricture")) +
  scale_fill_gradient(low="lightyellow",high = "red") +
  theme(legend.position = "none")
```

```
## Scale for 'fill' is already present. Adding another scale for 'fill', which
## will replace the existing scale.
```

```
str_3v3p_venn_plot
```



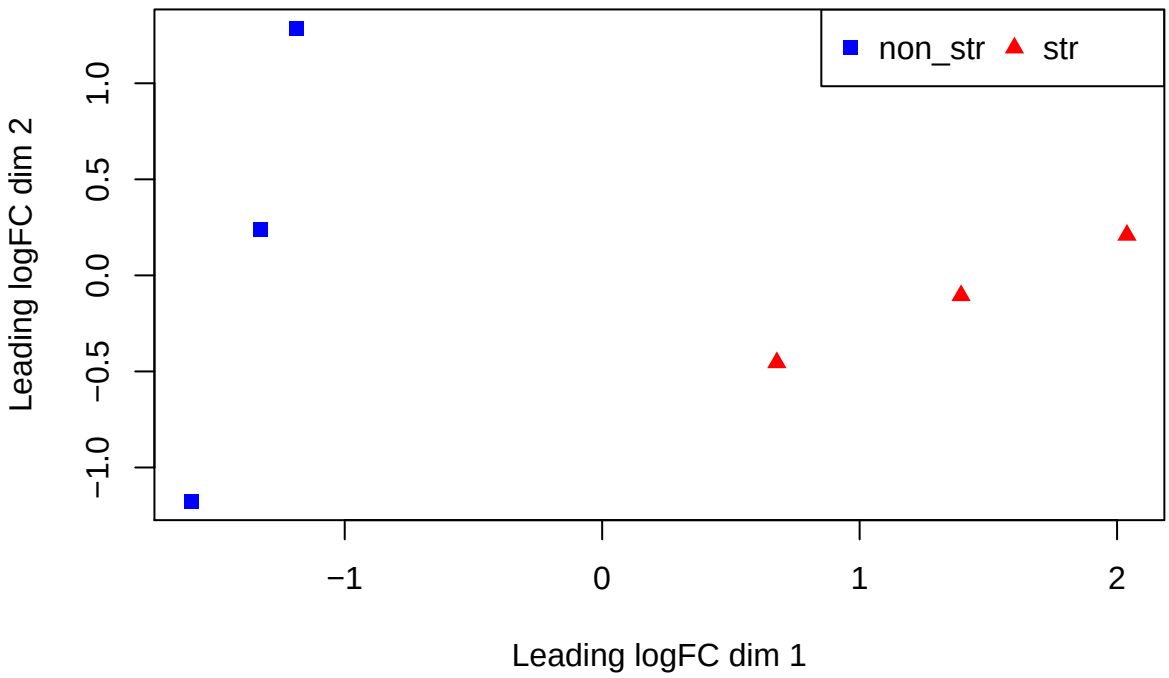

```
# ggsave(
#   "str_3v3p_venn.png",
#   str_3v3p_venn_plot,
#   width = 8.5,
#   height = 6,
#   dpi = 1200
# )
```

```
# 5 3 vs 3 post human ----------------------------------------------------

#tabulate
Duo_6h <- cbind(fc_3h$counts, fc_7h$counts, fc_9h$counts, fc_5h$counts,
```

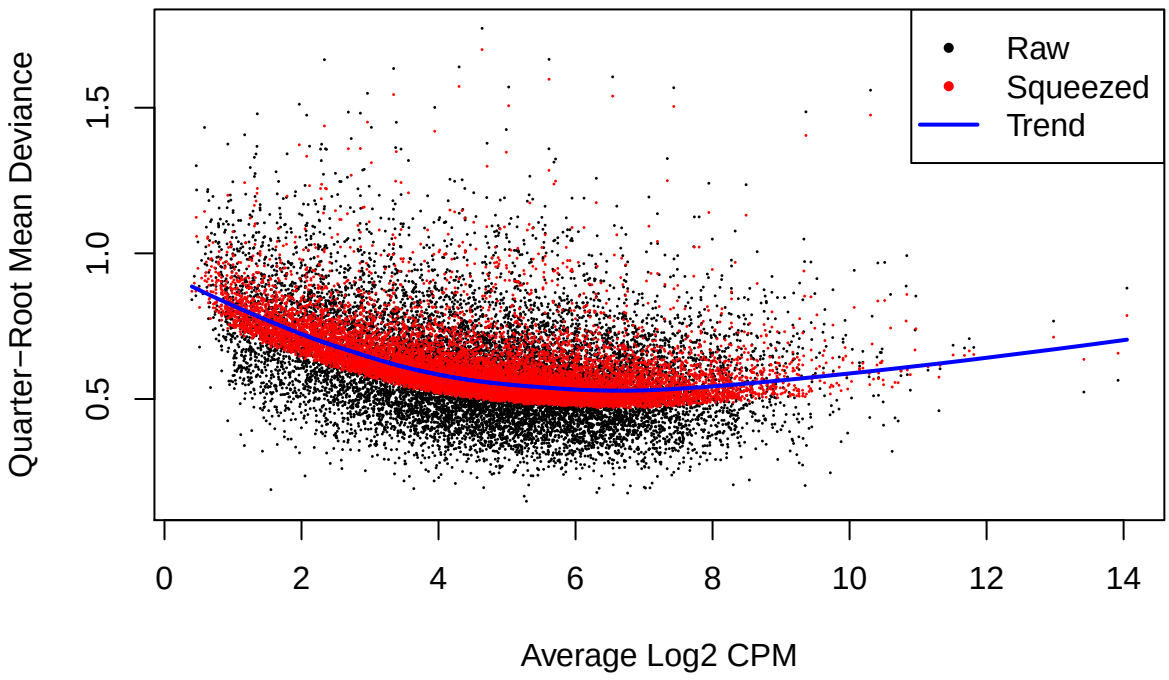
```
                fc_14h$counts, fc_21h$counts)
g_3v3ph <- c("duo_non","duo_non","duo_non","duo_str","duo_str","duo_str")
#trim lowly expressed exons
Duo_3v3ph <- DGEList(counts = Duo_6h[,1:6], group = g_3v3ph)
#match symbols to numbers
Symbol_3v3ph <- mapIds(org.Hs.eg.db, rownames(Duo_3v3ph), keytype = "ENTREZID", column="SYMBOL")
```

```
## 'select()' returned 1:1 mapping between keys and columns
```

```
#remove the ones that were NA
Duo_3v3ph <- Duo_3v3ph[!is.na(Symbol_3v3ph), ]
dim_Duo_3v3ph <- as.numeric(dim(Duo_3v3ph))
# remove all NA
Duo_3v3ph <- Duo_3v3ph[1:(dim_Duo_3v3ph[1]-1), ]

#counts per million have to have counts greater than X per million
#and be found in at least Y samples
round_3v3ph <- as.numeric(round(cpm(10, mean(Duo_3v3ph$samples$lib.size))))
#this value is used in the rowsum inequality
keep_3v3ph <-rowSums(cpm(Duo_3v3ph)>round_3v3ph) >= 2
Duo_3v3ph <- Duo_3v3ph[keep_3v3ph,]
# dim(Duo_3v3ph)

#TMM normalization
Duo_3v3ph.norm <-calcNormFactors(Duo_3v3ph, method = "TMM")
Duo_3v3ph.norm$samples

# write.table(Duo_3v3ph.norm$pseudo.counts, file = "str_3v3ph_pseudocounts.txt")

#data exploration of dataset
plotMDS(Duo_3v3ph.norm,pch = c(15,15,15,17,17,17),
        col = c("blue", "blue","blue", "red","red", "red","red","red"))
legend("topright", legend=c("non_str", "str"), pch=c(15,17), col=c("blue","red"), ncol=2)
```

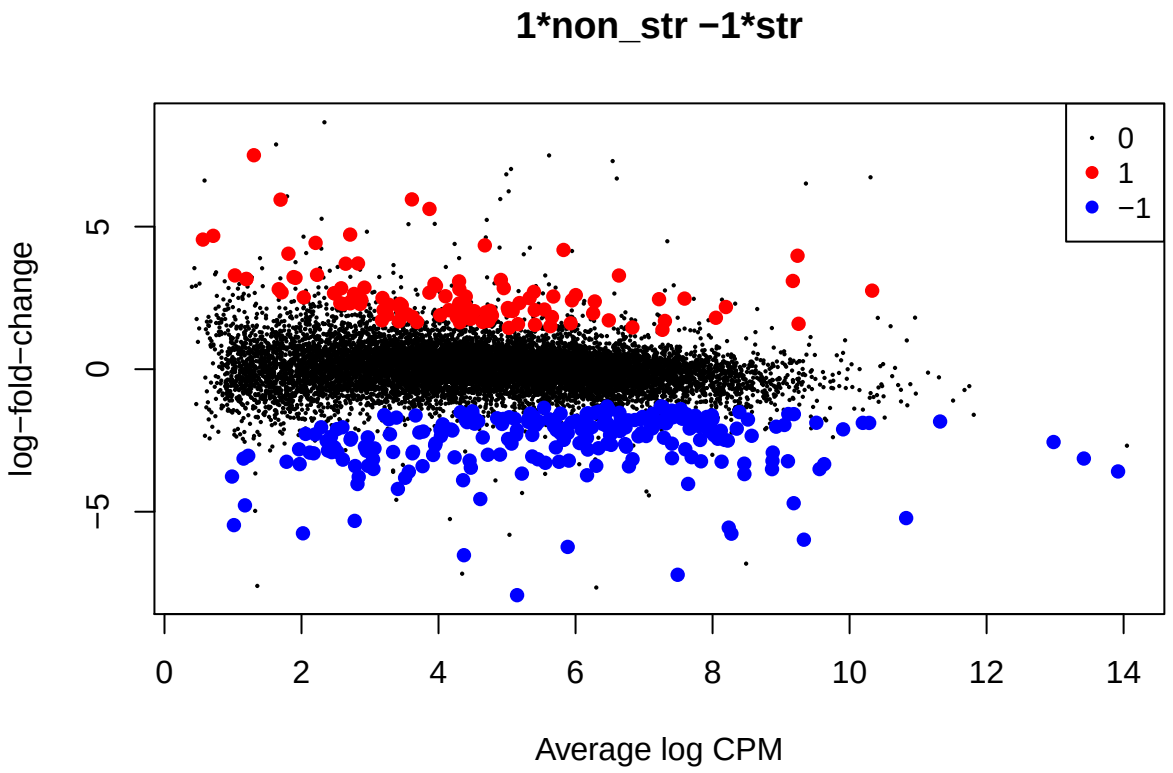



```
# dev.copy(png, paste0("str_3v3ph_plotMDS.png"))
# dev.off()

#Estimate common dispersion (could just return it to Duo_str.norm as it just adds columns)
des_3v3ph <- model.matrix(~ 0 + g_3v3ph)
colnames(des_3v3ph) <- c("non_str","str")
Duo_3v3ph.norm <- estimateCommonDisp(Duo_3v3ph.norm, des_3v3ph, verbose = TRUE, robust= TRUE)
# plotBCV(Duo_3v6.norm)

fit_3v3ph <- glmQLFit(Duo_3v3ph.norm, des_3v3ph, robust=TRUE)
head(fit_3v3ph$coefficients)
plotQLDisp(fit_3v3ph)
```




```
# dev.copy(png, paste0("str_3v3ph_plotQLDisp.png"))
# dev.off()
# summary(fit_3v3ph$df.prior)

# outliers from the mean-NB dispersion trend. Outliers are marked by small prior.df values:
o_3v3ph <- order(fit_3v3ph$df.prior)
Duo_3v3ph.norm$counts[o_3v3ph[1:6],] #genes tag used in the tutorial but our labels had names

con_3v3ph <- makeContrasts(non_str - str, levels=des_3v3ph)
res_3v3ph <- glmQLFTest(fit_3v3ph, contrast=con_3v3ph)
topTags(res_3v3ph)
#highly expressed in stricture than non-stricture

de_3v3ph <- decideTestsDGE(res_3v3ph, p.value=0.05)
# summary(de_3v3ph)
# plotSmear(res_3v3ph, de.tags=rownames(res_3v3ph)[de_3v3ph!=0])
# dev.copy(png, paste0("str_3v3ph_smear.png"))
# dev.off()

# write.table(topTags(res_3v3ph, n=100000)$table, file="str_3v3ph_topTags_glmQLF.txt")


# #imposing the 1.5 fold change
res.fc_3v3ph <- glmTreat(fit_3v3ph, contrast = con_3v3ph, lfc = log2(1.5))
topTags(res.fc_3v3ph)
de.fc_3v3ph <- decideTestsDGE(res.fc_3v3ph)
```

```
summary(de.fc_3v3ph)
plotMD(res.fc_3v3ph, status=de.fc_3v3ph, values=c(1,-1), col=c("red","blue"), legend="topright")
```




```
#Using goana
go_3v3ph <-goana(res_3v3ph, species = "Hs", FDR = 0.05)
# topGO(go_3v3ph, n=15)

#Supplementary table 6
topGO(go_3v3ph, n=15, sort="down")
# supplementary table 6a
write.table(topGO(go_3v3ph,n=1500, sort = "down"), file = "supplementary_table_6a_STR.txt")
topGO(go_3v3ph, n=15, sort="up")
#supplementary table 6b
write.table(topGO(go_3v3ph,n=2500, sort = "up"), file = "supplementary_table_6b_non_STR.txt")

go_3v3ph_str_dn <- rownames(topGO(go_3v3ph, n=15, sort="down"))
# for (i in 1:15){
#   write.table(getgeneid(go_3v3ph_str_dn[i],de_3v3ph),
# file = paste("go_3v3ph_str_dn",substr(go_3v3ph_str_dn[i],4,
# nchar(go_3v3ph_str_dn[i])),".txt",sep=""))
# }

go_3v3ph_str_up <- rownames(topGO(go_3v3ph, n=15, sort="up"))
# for (i in 1:15){
#   write.table(getgeneid(go_3v3ph_str_up[i],de_3v3ph),
```

```
# file = paste("go_3v3ph_str_up",substr(go_3v3ph_str_up[i],4,
# nchar(go_3v3ph_str_up[i])),".txt",sep=""))
# }

# Table 2 Tope Gene Ontology sets in Stricture and non-Stricture
write.table(topGO(go_3v3ph, n=15, sort="up", ontology = "BP"), file = "Table2_nonstr.txt")
write.table(topGO(go_3v3ph, n=15, sort="down", ontology = "BP"), file = "Table2_str.txt")
# end table 2

# Using human entrez ids to look at pathways

topTags_3v3ph <- topTags(res_3v3ph, n=100000)$table
#neg
negFC_3v3ph <- topTags_3v3ph[,1]<0
neg_gene_3v3ph <- as.numeric(topTags_3v3ph[,5][negFC_3v3ph])
names(neg_gene_3v3ph) <-rownames(topTags_3v3ph)[negFC_3v3ph]
GOdata_neg_3v3ph <- new("topGOdata", description = "Significant GO",
                        ontology = "BP", allGenes = neg_gene_3v3ph, geneSel = topDiffGenes,
                        nodeSize = 10, annot = annFUN.org ,
                        mapping = "org.Hs.eg.db", ID="entrez")
```

```
##
## Building most specific GOs .....

##  ( 8644 GO terms found. )

##
## Build GO DAG topology ..........

##  ( 12716 GO terms and 29893 relations. )

##
## Annotating nodes ...............

##  ( 5739 genes annotated to the GO terms. )
```

```
resultFisher_neg_3v3ph <- runTest(GOdata_neg_3v3ph, algorithm = "classic", statistic = "fisher")
```

```
##
##           -- Classic Algorithm --
##
##       the algorithm is scoring 2009 nontrivial nodes
##       parameters:
##           test statistic: fisher
```

```
resultKS_neg_3v3ph <- runTest(GOdata_neg_3v3ph, algorithm = "classic", statistic = "ks")
```

```
##
##           -- Classic Algorithm --
##
```

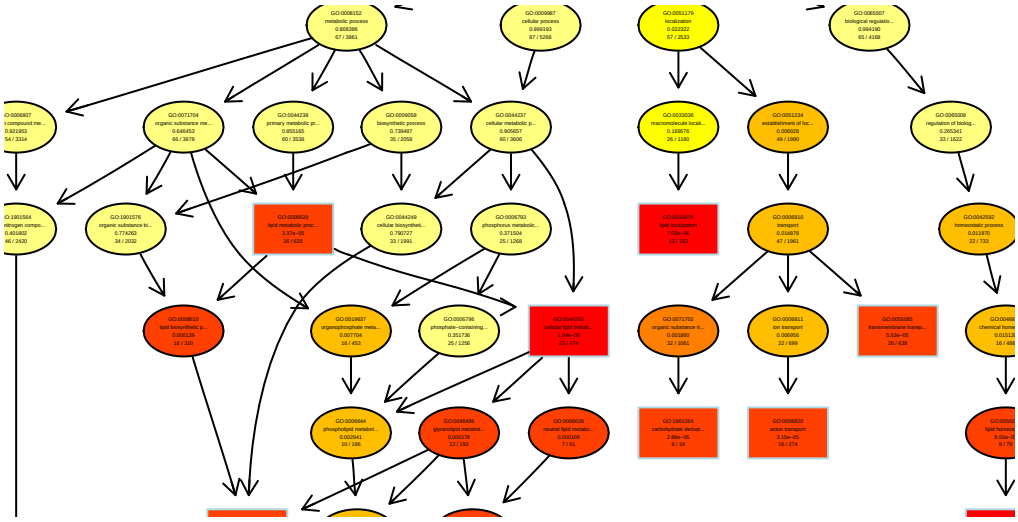
```
##       the algorithm is scoring 4257 nontrivial nodes
##       parameters:
##           test statistic: ks
##           score order: increasing
```

```
resultKS.elim_neg_3v3ph <- runTest(GOdata_neg_3v3ph, algorithm = "elim", statistic = "ks")
```

```
##
##           -- Elim Algorithm --
##
##       the algorithm is scoring 4257 nontrivial nodes
##       parameters:
##           test statistic: ks
##           cutOff: 0.01
##           score order: increasing

##
##   Level 19:  1 nodes to be scored    (0 eliminated genes)

##
##   Level 18:  1 nodes to be scored    (0 eliminated genes)

##
##   Level 17:  4 nodes to be scored    (0 eliminated genes)

##
##   Level 16:  10 nodes to be scored   (0 eliminated genes)

##
##   Level 15:  26 nodes to be scored   (0 eliminated genes)

##
##   Level 14:  49 nodes to be scored   (12 eliminated genes)

##
##   Level 13:  91 nodes to be scored   (157 eliminated genes)

##
##   Level 12:  154 nodes to be scored  (181 eliminated genes)

##
##   Level 11:  275 nodes to be scored  (202 eliminated genes)

##
##   Level 10:  429 nodes to be scored  (282 eliminated genes)

##
##   Level 9:   561 nodes to be scored  (399 eliminated genes)

##
##   Level 8:   599 nodes to be scored  (575 eliminated genes)
```

```
##
##   Level 7:   689 nodes to be scored  (898 eliminated genes)

##
##   Level 6:   613 nodes to be scored  (1333 eliminated genes)

##
##   Level 5:   409 nodes to be scored  (2024 eliminated genes)

##
##   Level 4:   222 nodes to be scored  (2177 eliminated genes)

##
##   Level 3:   103 nodes to be scored  (2434 eliminated genes)

##
##   Level 2:   20 nodes to be scored   (2446 eliminated genes)

##
##   Level 1:   1 nodes to be scored    (2446 eliminated genes)
```

```
allRes_neg_3v3ph <- GenTable(GOdata_neg_3v3ph, classicFisher = resultFisher_neg_3v3ph,
                             classicKS = resultKS_neg_3v3ph, elimKS = resultKS.elim_neg_3v3ph,
                             orderBy = "classicFisher", ranksOf = "classicFisher", topNodes = 10)


#supp fig 6a
#resembles the GOANA data
showSigOfNodes(GOdata_neg_3v3ph, score(resultFisher_neg_3v3ph), firstSigNodes = 10, useInfo = 'all')
```



```
printGraph(GOdata_neg_3v3ph, resultFisher_neg_3v3ph, firstSigNodes = 10, fn.prefix = "SuppFig6a",
           useInfo = "all", pdfSW = TRUE)

#pos
posFC_3v3ph <- topTags_3v3ph[,1]>0
pos_gene_3v3ph <- as.numeric(topTags_3v3ph[,5][posFC_3v3ph])
names(pos_gene_3v3ph) <-rownames(topTags_3v3ph)[posFC_3v3ph]
GOdata_pos_3v3ph <- new("topGOdata", description = "Significant GO",
                        ontology = "BP", allGenes = pos_gene_3v3ph, geneSel = topDiffGenes,
                        nodeSize = 10, annot = annFUN.org , mapping = "org.Hs.eg.db", ID="entrez")
```

```
##
## Building most specific GOs .....

##  ( 8976 GO terms found. )

##
## Build GO DAG topology ..........

##  ( 13204 GO terms and 31114 relations. )

##
## Annotating nodes ...............

##  ( 6333 genes annotated to the GO terms. )
```

```
resultFisher_pos_3v3ph <- runTest(GOdata_pos_3v3ph, algorithm = "classic", statistic = "fisher")
```

```
##
##           -- Classic Algorithm --
##
##       the algorithm is scoring 1354 nontrivial nodes
##       parameters:
##           test statistic: fisher
```

```
resultKS_pos_3v3ph <- runTest(GOdata_pos_3v3ph, algorithm = "classic", statistic = "ks")
```

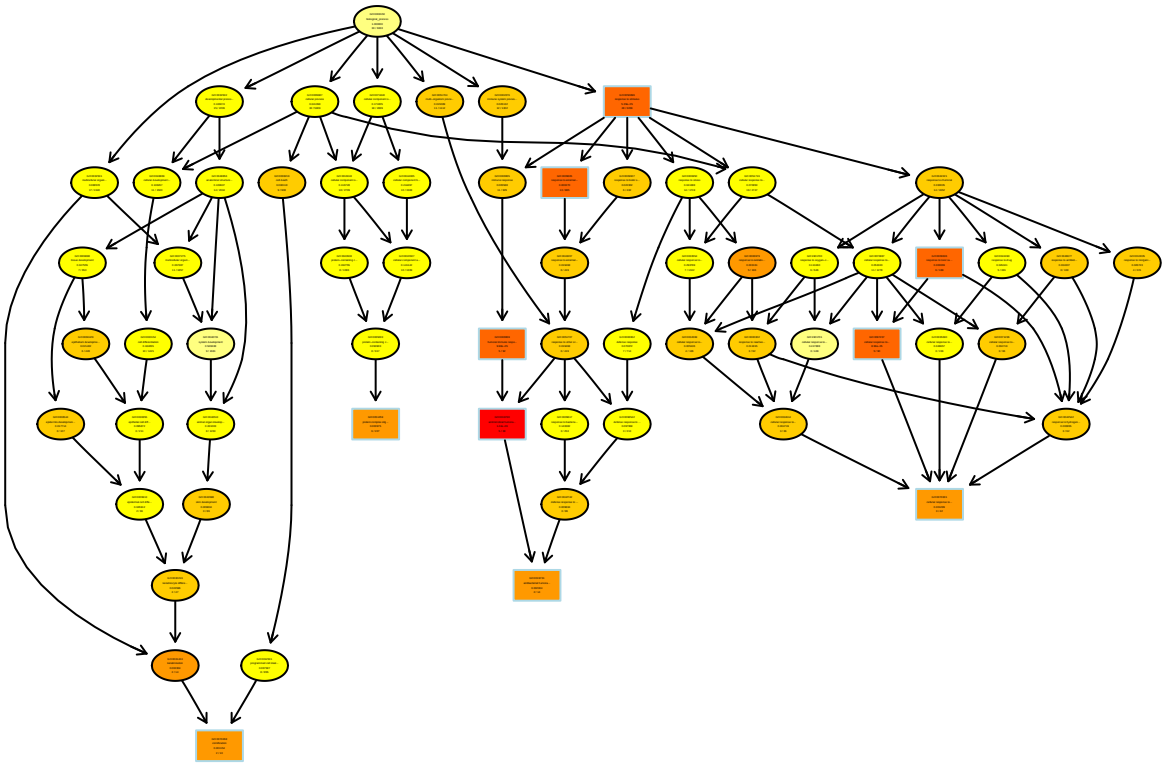
```
##
##           -- Classic Algorithm --
##
##       the algorithm is scoring 4537 nontrivial nodes
##       parameters:
##           test statistic: ks
##           score order: increasing
```

```
resultKS.elim_pos_3v3ph <- runTest(GOdata_pos_3v3ph, algorithm = "elim", statistic = "ks")
```

```
##
##           -- Elim Algorithm --
##
##       the algorithm is scoring 4537 nontrivial nodes
##       parameters:
##           test statistic: ks
##           cutOff: 0.01
##           score order: increasing

##
##   Level 17:  3 nodes to be scored    (0 eliminated genes)

##
##   Level 16:  11 nodes to be scored   (0 eliminated genes)

##
##   Level 15:  22 nodes to be scored   (0 eliminated genes)

##
##   Level 14:  65 nodes to be scored   (27 eliminated genes)

##
##   Level 13:  134 nodes to be scored  (102 eliminated genes)

##
##   Level 12:  196 nodes to be scored  (706 eliminated genes)

##
##   Level 11:  307 nodes to be scored  (1165 eliminated genes)
```

```
##
##   Level 10:  473 nodes to be scored  (1309 eliminated genes)

##
##   Level 9:   618 nodes to be scored  (1972 eliminated genes)

##
##   Level 8:   632 nodes to be scored  (2377 eliminated genes)

##
##   Level 7:   702 nodes to be scored  (2909 eliminated genes)

##
##   Level 6:   629 nodes to be scored  (3288 eliminated genes)

##
##   Level 5:   400 nodes to be scored  (3730 eliminated genes)

##
##   Level 4:   222 nodes to be scored  (4040 eliminated genes)

##
##   Level 3:   102 nodes to be scored  (4069 eliminated genes)

##
##   Level 2:   20 nodes to be scored   (4120 eliminated genes)

##
##   Level 1:   1 nodes to be scored    (4120 eliminated genes)
```

```
allRes_pos_3v3ph <- GenTable(GOdata_pos_3v3ph, classicFisher = resultFisher_pos_3v3ph,
                             classicKS = resultKS_pos_3v3ph, elimKS = resultKS.elim_pos_3v3ph,
                             orderBy = "classicFisher", ranksOf = "classicFisher", topNodes = 10)

#supp figure 7
#resembles the GOANA data
showSigOfNodes(GOdata_pos_3v3ph, score(resultFisher_pos_3v3ph), firstSigNodes = 10, useInfo = 'all')
```



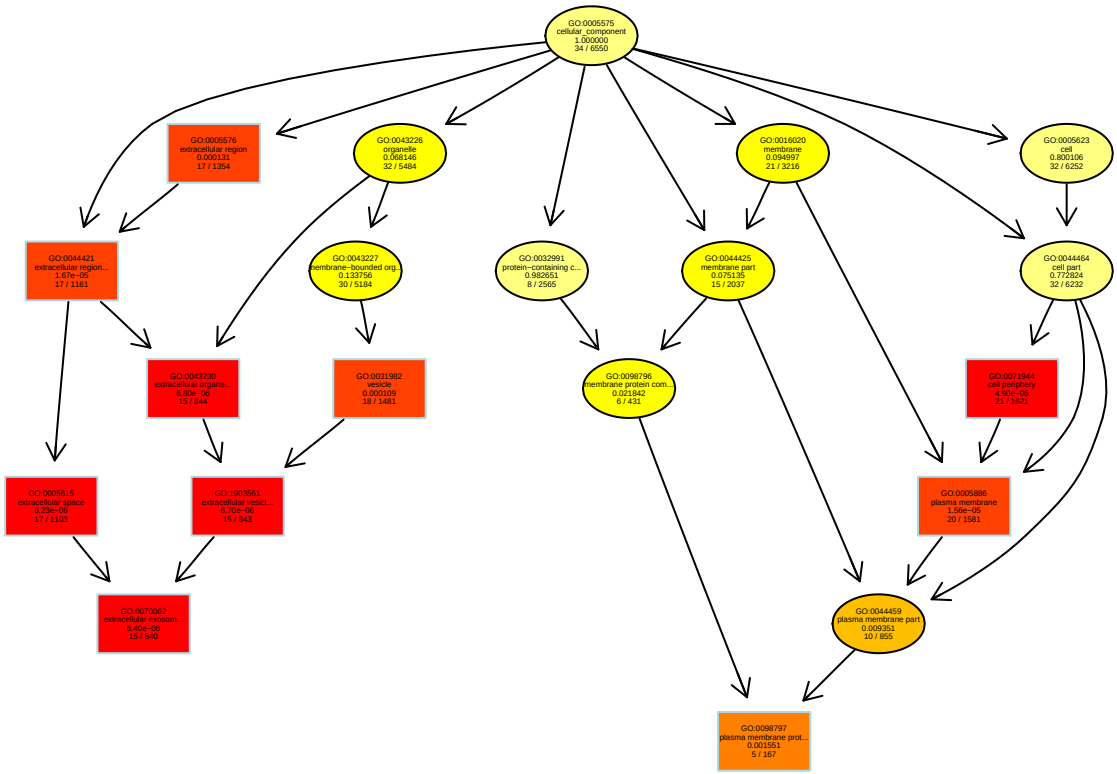
```
printGraph(GOdata_pos_3v3ph, resultFisher_pos_3v3ph, firstSigNodes = 10,
           fn.prefix = "SuppFig7", useInfo = "all", pdfSW = TRUE)

# supp figure 6b
### redo for CC
GOdata_CC_pos_3v3ph <- new("topGOdata", description = "Significant GO", ontology = "CC",
                           allGenes = pos_gene_3v3ph, geneSel = topDiffGenes, nodeSize = 10,
                           annot = annFUN.org , mapping = "org.Hs.eg.db", ID="entrez")
```

```
##
## Building most specific GOs .....

##  ( 1420 GO terms found. )

##
## Build GO DAG topology ..........

##  ( 1736 GO terms and 3264 relations. )

##
## Annotating nodes ...............

##  ( 6550 genes annotated to the GO terms. )
```

```
resultFisher_CC_pos_3v3ph <- runTest(GOdata_CC_pos_3v3ph, algorithm = "classic", statistic = "fisher")
```

```
##
##           -- Classic Algorithm --
##
##       the algorithm is scoring 224 nontrivial nodes
##       parameters:
##           test statistic: fisher
```

```
showSigOfNodes(GOdata_CC_pos_3v3ph, score(resultFisher_CC_pos_3v3ph), firstSigNodes = 10,
               useInfo = 'all')
```




```
printGraph(GOdata_CC_pos_3v3ph, resultFisher_CC_pos_3v3ph, firstSigNodes = 10,
           fn.prefix = "SuppFig6b", useInfo = "all", pdfSW = TRUE)

GOdata_CC_neg_3v3ph <- new("topGOdata", description = "Significant GO", ontology = "CC",
                           allGenes = neg_gene_3v3ph, geneSel = topDiffGenes, nodeSize = 10,
                           annot = annFUN.org , mapping = "org.Hs.eg.db", ID="entrez")
```

```
##
## Building most specific GOs .....
```

```
##  ( 1253 GO terms found. )
```

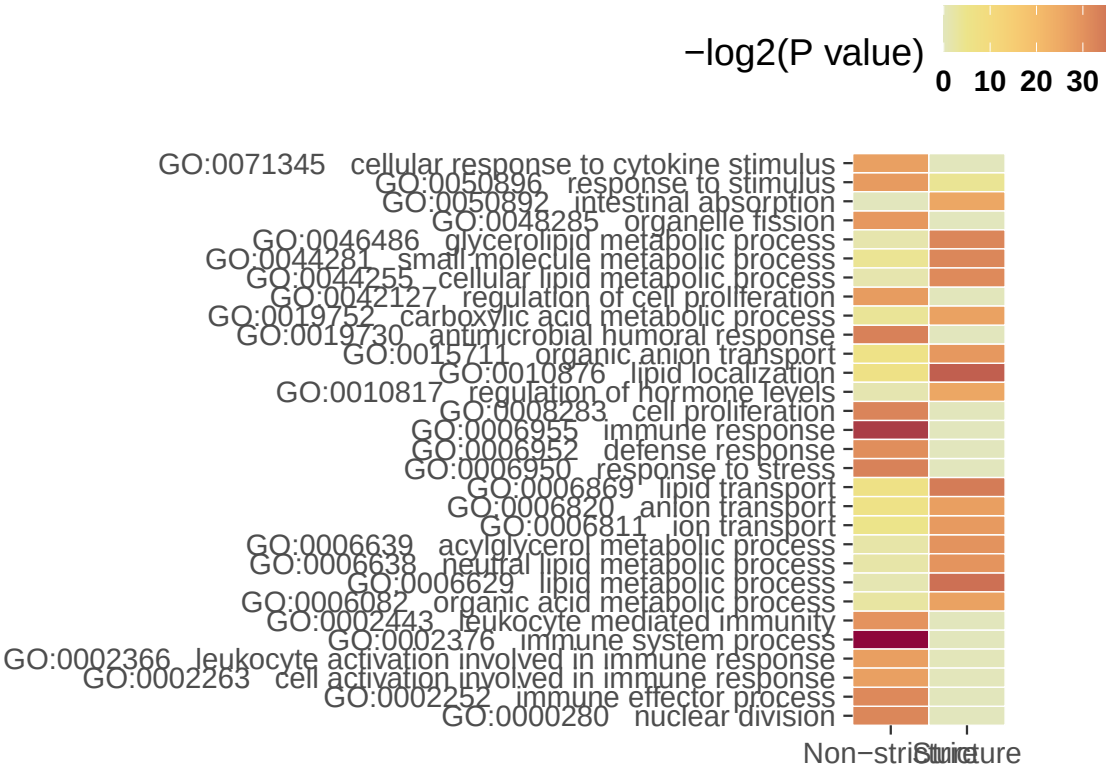
```
##
## Build GO DAG topology ..........

##  ( 1567 GO terms and 2960 relations. )

##
## Annotating nodes ...............

##  ( 5950 genes annotated to the GO terms. )
```

```
resultFisher_CC_neg_3v3ph <- runTest(GOdata_CC_neg_3v3ph, algorithm = "classic", statistic = "fisher")
```

```
##
##           -- Classic Algorithm --
##
##       the algorithm is scoring 282 nontrivial nodes
##       parameters:
##           test statistic: fisher
```

```
# showSigOfNodes(GOdata_CC_neg_3v3ph, score(resultFisher_CC_neg_3v3ph),
#firstSigNodes = 10, useInfo = 'all')
# printGraph(GOdata_CC_neg_3v3ph, resultFisher_CC_neg_3v3ph, firstSigNodes = 10,
#fn.prefix = "neg_Fisher_CC_3v3ph_tGO", useInfo = "all", pdfSW = TRUE)



#heatmap table
tbl_up_bp_3v3ph <- topGO(go_3v3ph, n=15, sort="up", ontology = "BP")
tbl_dn_bp_3v3ph <- topGO(go_3v3ph, n=15, sort="down", ontology = "BP")
log.tbl_up_bp_3v3ph <- append(-log2(tbl_up_bp_3v3ph[,6]),-log2(tbl_dn_bp_3v3ph[,6]))
log.tbl_dn_bp_3v3ph <- append(-log2(tbl_up_bp_3v3ph[,7]),-log2(tbl_dn_bp_3v3ph[,7]))

rowname <- rep(c(paste(rownames(tbl_up_bp_3v3ph),
                       " ",tbl_up_bp_3v3ph$Term),paste(rownames(tbl_dn_bp_3v3ph),
                                                       " ",tbl_dn_bp_3v3ph$Term)),2)
colname <- c(rep("Non-stricture",30),rep("Stricture",30))
P_value <- append(log.tbl_up_bp_3v3ph, log.tbl_dn_bp_3v3ph)

#create dataframe and remove 0's
log.tbl_all_bp_3v3ph <- as.data.frame(cbind(rowname,colname,as.numeric(P_value)))
log.tbl_all_bp_3v3ph[log.tbl_all_bp_3v3ph == 0] <- NA
# write.table(log.tbl_all_bp_3v3ph, file = "log_tbl_all_str_3v3ph.txt")

# supplementary Fig 4b
str_3v3ph_heatmap_GO <- ggplot(log.tbl_all_bp_3v3ph,
                               mapping = aes(x = colname, y = rowname, fill = P_value)) +
  geom_tile(colour="white",size=0.25) +
  scale_y_discrete(expand=c(0,0)) +
  scale_x_discrete(expand=c(0,0)) +
  theme_grey(base_size = 14) +
  labs(x = "", y = "") +
  coord_fixed(ratio=0.25) +
```

```
  theme(
    #bold font for legend text
    legend.text=element_text(face="bold"),
    legend.position="top",
    #set thickness of axis ticks
    axis.ticks=element_line(size=0.4),
    #remove plot background
    plot.background=element_blank(),
    #remove plot border
    panel.border=element_blank()) +
  scale_fill_continuous_sequential(palette = "Heat", name="-log2(P value)")
str_3v3ph_heatmap_GO
```




```
# ggsave(
#   "str_3v3ph_heatmap_GO_BP.png",
#   str_3v3ph_heatmap_GO,
#   width = 8.5,
#   height = 11,
#   dpi = 1200
# )
```