RNAseq

Alessia Squitieri

2/10/2020

Contents

Introduction

RNA-seq data is a recent approach to carry out expression profiling using high-throughput (HTS) technologies. The aim of this project is to perform a complete RNAseq analysis with Bioconductor's packages, comparing individuals having early vs advance tumoral stages.

Methods

In this study we have analysed a Ranged Summarized Experiment (RSE) from the TCGA tab - The Cancer Genome Atlas - available in https://jhubiostatistics.shinyapps.io/recount/. The object that we have used is RSE of pleura's cancer. First of all, we have downloaded the data and we have opened the file into R with the function "load". Exploring our data, we noticed that we are working with a dataset of 58037 rows and 87 columns. In ranged summarized experiments'objects, rows represent genomic ranges of interest, accesibles trought the function "rowRanges". We have used this function to create the variable "annot" in which there are 58037 ranges and 3 metadata columns: "gene_id", "bp_length" and "symbol". Exploring the Bioconductor's vignette of SummarizedExperiment we have been able to create the variable "counts" using the function "assay(rse_gene,"counts")". Counts is a matrix that contains the reads (rows) aligned and paired with the corresponding samples.

The first step of our analysis is the normalization. Normalization is a crucial point of RNAseq, we know that the number of read is directly correlated to the sequence depth, the transcript length and also the GC content. In our case we have tested three different type of normalization, RPKM, TMM and CQN, only CQN takes into account the GC content so, usually, is the best option. So, we have tested the three different type of normalization and then we have used the function maPlot to visualize the results. Looking at the plots is evident that, in our study, the best choice is to use the TMM normalization. In fact, the plots used to check the normalization of our reads, show the relationship between the mean and the variance. In an ideal plot we expect to visualize that the red line, indicating the mean and variance relationship, is around zero. In the case of CQN and RPKM normalization we have not this situation; we have decided to consider the counts normalized with TMM. We have continued the analysis using the counts.tmm object that has been filtered with the function "filterCounts(counts.tmm)" and we obtained the new variable counts.f. Looking at the dimension of the two counts we could notice how much the filter has reduced the number of counts (58037 before filtering, 28158 later).

The second step has been creating the variable GROUP, containing the individuals having early vs advance tumoral stages. We have used the code available in the campus virtual and we have "extract" the variable with the function "GROUP <- rse_gene$GROUP", then we have transformed the variable into a factor with two levels "early" and "late" and we have checked that there are not NA. Finally we have used the function table to check the contents: 26 individuals with early tumoral stage and 61 with late tumoral stage.

Now we could proceed with the differential expression analysis. What's the goal of this step? We have to detect genes that are differentially expressed but we have to take in mind that we are working with

count data. In RNAseq is used the negative binomial distribution because it takes into account the overdispersion, considering two parameters: the mean (lambda) and the overdispersion, the ratio of mean and variance (phi). There are some packages in R that are used to calculate overdispersion. In our case we have used egdeR package that assumes two type of overdispersion: common and tagwise. First of all, we have created the DGElist object that contains counts (in our case counts.f that is our normalizated and filtered counts) and the variable GROUP that, as said before, is a factor, with length equal to the number of columns of counts, denoting the experimental group (early vs late). EgdeR provides two ways of estimating the dispersion(s), the quantile-adjusted conditional maximum likelihood (qCML) method and the Cox-Reid profile-adjusted likelihood (CR) method. In general, we apply the qCML method to experiments with single factor and the CR method to experiments with multiple factors. Therefore,the qCML method (i.e. the estimateCommonDisp() and estimateTagwiseDisp() function) is recommended for a study with single factor. (edgeR: differential expression analysis of digital gene expression data, M. Robinson, D. McCarthy & Y. Chen/https://rnajournal.cshlp.org/content/suppl/2013/04/02/rna.039271.113.DC1/edgeR.pdf). In our case the best option has been to perform a qCML method, estimating the common and tagwise dispersion. We used the function "estimateCommonDisp" and "estimateTagwiseDisp", using the DGElist that we have had previously created as object of the function. We obtained a common dispersion of 0.87 that we stored in the variable CD_parameter and we created a dataframe containing the values of the tagwise dispersion for each gene. Then we applied the exact test, the results of the NB exact test can be accessed conveniently using the topTags function applied to the object produced by exactTest. The topTags shows that there is differential expression between early and late groups; we have used the function "sum(topTags(edgeR_common, n=Inf)$table$FDR < 0.05)" and "sum(topTags(edgeR_TGW, n=Inf)$table$FDR < 0.05)" to estimate the number of reads that have a FDR < 0.05 and we it returned that 2216 are significantlly expressed with the common dispersion and 661 with the tagwise dispersion. We can visualize the two type of dispersion using the plotBCV(). To visualize the DE - with only the common dispersion - we used the fold-change plot (the log-fold change against the log-concentration for each tag), with the top 500 most DE counts highlighted in red and we observed that the red dots fall outside the blue lines so the reads identified by edgeR as differentially expressed are truly differentially expressed. Then, we applied another filter using the function using the "mask" in which we selected only genes with FDR < 0.05 and logFC > 1, and we obtained 2048 reads from the DE_genes_CD and 620 from DE_genes_TGW variables. Then, we used a function to define if the DE of the tags is a down or up regulation, based on the value of the logFC (<0 down regulation and >0 up regulation). Finally we defined the variable "geneUniverse" in which are stored all the reads that were previously selected in the counts.f.

We proceed with the last step of our project, the enrichment analysis. This is our main goal, check if there is any pathway within our list of DE tags. In fact, we obtained a huge number of DE genes but now we want to check which pathways the differentially expressed genes are implicated in. First of all, we performed the annotation of the selected genes with common and tagwise dispersion and also of the geneUniverse variables. For the annotation we used the biomaRt package with the function "getBM", obtaining the corresponding ENTREZ ID from our ENSEMBL ID. Well, we obtained the genes annotated and we can see that the number of them is reduced a lot (330) in the case of DE with common dispersion, 113 in the case of DE with tagwise dispersion and 3938 in the case of geneUniverse. We used the cluster profiler package with the function enrichGO that use the Gene Ontology browser, specifying the gene set of interest, the universe (geneUniverse), the type of pvalue adjust (we used bonferroni and fdr and we compared the difference), the pvalue cutoff and the qvalue cutoff, that we put at 0.05. We performed the analysis of DE of both tagwise and common dispersion using two different types of correction.

Results

We performed enrichment analysis with bonferroni and fdr correction and they give us two different results that we visualized through the plots. In the first case, using bonferroni correction and DE genes calculted with common dispersion we obtained 10 enriched terms, looking at the description we can notice that there are basically two main function that these genes are involed in: the contraction and the neuronal transmission, suggesting a strong relationship between the late stage of pleura cancer and an overexpression of genes involved in these pathways. With the same correction but using the DEG of tagwise dispersion the enriched terms are 8 and also in this case are present pathways related to the contraction but there are not gene natworks involved in the neuronal transmission. On the other hand, are showed new pathways:

supramolecular complex, supramolecular polymer and supramolecular fiber. With the FDR correction, as we aspected, the number of enriched terms returned by the analysis is higher: 19 in the case of DEG with common dispersion and 18 using the DE tags with tagwise dispersion. Let's look at the description. In these cases there are always the pathways involved in the contraction and in neuronal transmission but there are also showed others type of overexpressed genes networks as cation channel complex, transporter complex, DNA packaging complex and nucleosome. Making some research on internet, we have discovered that other studies in which has been performed GO analysis, showed significant overexpression of genes in cancer of pleura, reflecting several important biological functions, among them DNA replication and repair as well as cytoskeleton organization and biogenesis. In our case, in particular we found a strong influence of supramolecular complex, polymer and fiber all involved in cytoskeletron assembly. Another important issue that we found is a correlation between A3 receptors'expression and malignant cancer of pleura. A3 receptors are involved in neuronal transmission pathway that we can see in our gene networks.

Appendix

```
############### LOAD THE DATA #################
setwd("~/Desktop/BIOINFORMATICS ")
library(Biobase)
library(DESeq2)
load("rse_gene_pleura.Rdata")
annot <- rowRanges(rse_gene) # GET INFO
counts <- assay(rse_gene, "counts") # number of reads overlapping a genomic position
dim(counts)
```

```
## [1] 58037    87
```

```
############### NORMALIZATION #################
# 1. RPKM normalization

width <- annot$bp_length
counts.rpkm <- t(t(counts / width *1000)
                 /colSums(counts)*1e6)
counts.rpkm[1:5,1:4]
```

```
##                    E6088A41-B0DC-4FBF-8D14-BE78024CF8CD
## ENSG00000000003.14                           6.373345551
## ENSG00000000005.5                            0.009630659
## ENSG00000000419.12                          14.293267986
## ENSG00000000457.13                           1.116543251
## ENSG00000000460.16                           0.947944983
##                    F569915C-8F77-4D67-9730-30824DB57EE5
## ENSG00000000003.14                             5.232373
## ENSG00000000005.5                              0.000000
## ENSG00000000419.12                            27.940235
## ENSG00000000457.13                             2.820183
## ENSG00000000460.16                             2.596574
##                    E3B71CB7-673B-4741-8607-4F0A11633036
## ENSG00000000003.14                            4.4927569
## ENSG00000000005.5                             0.0174287
## ENSG00000000419.12                           27.3492653
## ENSG00000000457.13                            1.5984669
## ENSG00000000460.16                            2.6127310
##                    DAF84798-FE3F-403C-B589-7F256AF752BE
## ENSG00000000003.14                           1.694402331
```

```
## ENSG00000000005.5                          0.002541288
## ENSG00000000419.12                        32.087592868
## ENSG00000000457.13                         1.661317764
## ENSG00000000460.16                         1.773561542
```

```r
# 2. TMM normalization
library(tweeDEseq)
counts.tmm <- normalizeCounts(counts, method="TMM")
counts.tmm[1:5,1:4]
```

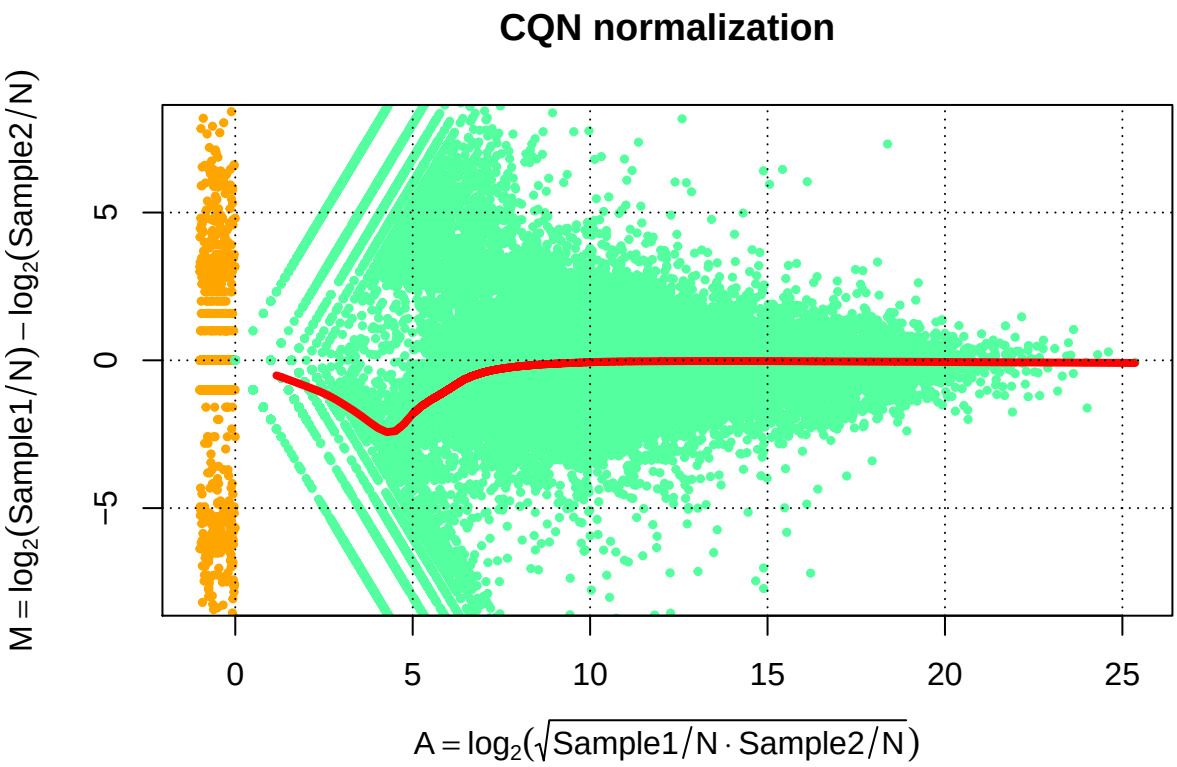
```
##                    E6088A41-B0DC-4FBF-8D14-BE78024CF8CD
## ENSG00000000003.14                               190362
## ENSG00000000005.5                                   102
## ENSG00000000419.12                               113625
## ENSG00000000457.13                                50616
## ENSG00000000460.16                                37254
##                    F569915C-8F77-4D67-9730-30824DB57EE5
## ENSG00000000003.14                               128726
## ENSG00000000005.5                                     0
## ENSG00000000419.12                               182948
## ENSG00000000457.13                               105304
## ENSG00000000460.16                                84052
##                    E3B71CB7-673B-4741-8607-4F0A11633036
## ENSG00000000003.14                               128772
## ENSG00000000005.5                                   177
## ENSG00000000419.12                               208633
## ENSG00000000457.13                                69536
## ENSG00000000460.16                                98533
##                    DAF84798-FE3F-403C-B589-7F256AF752BE
## ENSG00000000003.14                                44405
## ENSG00000000005.5                                    24
## ENSG00000000419.12                               223813
## ENSG00000000457.13                                66080
## ENSG00000000460.16                                61157
```

```r
# 3. CQN normalization
## we have to obtain the gc content from ensembl:
library(biomaRt)
mart  <- useEnsembl(biomart="ensembl",
                    dataset="hsapiens_gene_ensembl",
                    host = "uswest.ensembl.org", mirror = "www")

gc_content <- getBM(attributes=c("ensembl_gene_id_version",
                                 "percentage_gene_gc_content"),
                    filters="ensembl_gene_id_version",
                    values=rownames(counts),
                    mart=mart)

## now we can create a dataframe containing our variable of interest:
genes <- intersect(rownames(counts), gc_content$ensembl_gene_id_version) # 38321 genes
genes_length  <- unlist(list(annot[genes,]$bp_length), use.names = FALSE)
annotation_gc <- (cbind(genes_length, gc_content))
annotation_gc_2 <- annotation_gc[, c(1,3)]
```

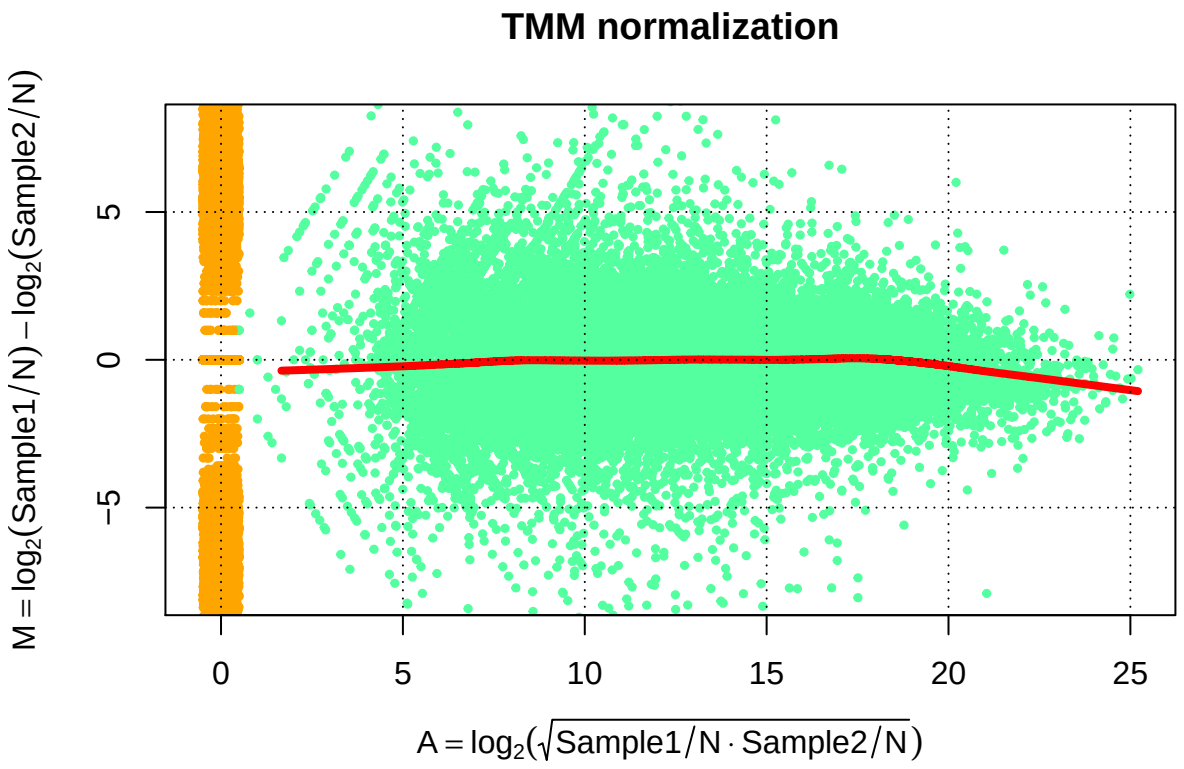
```
# we have to remove the id of genes from column and put it as rawnames
colnames(annotation_gc_2) <- c( "genes_length", "percentage_gene_gc_content")
rownames(annotation_gc_2) <- annotation_gc$ensembl_gene_id_version

#Now we have the correct form of dataframe and we cab apply the cqn normalization:
library(cqn)
counts.cqn <- normalizeCounts(counts, method="cqn", annot = annotation_gc_2)
```

```
## RQ fit ..........................................................................................
## SQN .
```

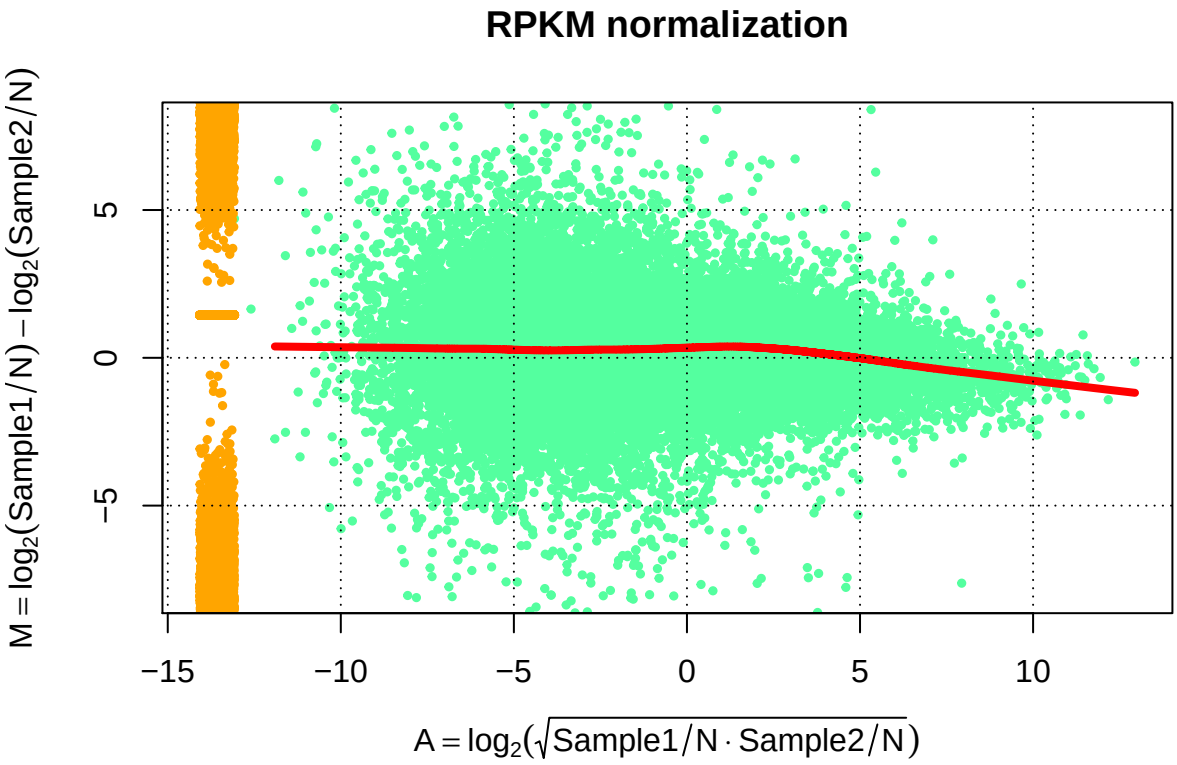
```
########## VISUALIZZATION OF NORMALIZATION ############
library(edgeR)
# 1. CQN
maPlot(counts.cqn[,1], counts.cqn[,2], pch=19, cex=.5, ylim=c(-8,8),
       allCol="seagreen1", lowess=TRUE,
       xlab=expression( A == log[2] (sqrt(Sample1/N %.% Sample2/N)) ),
       ylab=expression(M == log[2](Sample1/N)-log[2](Sample2/N)))
grid(col="black")
title("CQN normalization")
```




```
# 2. TMM
maPlot(counts.tmm[,1], counts.tmm[,2], pch=19, cex=.5, ylim=c(-8,8),
       allCol="seagreen1", lowess=TRUE,
       xlab=expression( A == log[2] (sqrt(Sample1/N %.% Sample2/N)) ),
       ylab=expression(M == log[2](Sample1/N)-log[2](Sample2/N)))
grid(col="black")
title("TMM normalization")
```




```
# 3. RPKM
maPlot(counts.rpkm[,1], counts.rpkm[,2], pch=19, cex=.5, ylim=c(-8,8),
       allCol="seagreen1", lowess=TRUE,
       xlab=expression( A == log[2] (sqrt(Sample1/N %.% Sample2/N)) ),
       ylab=expression(M == log[2](Sample1/N)-log[2](Sample2/N)))
grid(col="black")
title("RPKM normalization")
```




```
######### filter counts
counts.f <- filterCounts(counts.tmm)
dim(counts.f)
```

```
## [1] 28158    87
```

```
################################## EXTRACTING THE VARIABLE GROUP #################
stage <- rse_gene$gdc_cases.diagnoses.tumor_stage

ids.early <- grep(paste("stage i$", "stage ia$", "stage ib$" ,
                        "stage ic$", "stage ii$", "stage iia$",
                        "stage iib$" , "stage iic$",
                        sep="|"), stage) # 26

ids.late <- grep(paste("stage iii$", "stage iiia$", "stage iiib$" ,
                       "stage iiic$", "stage iv$", "stage iva$",
                       "stage ivb$" , "stage ivc$",
                       sep="|"), stage) # 61

colData(rse_gene)$GROUP <- rep(NA, ncol(rse_gene))
colData(rse_gene)$GROUP[ids.early] <- "early"
colData(rse_gene)$GROUP[ids.late] <- "late"

GROUP <- rse_gene$GROUP
GROUP <- as.factor(GROUP)

table(GROUP)
```

```
## GROUP
```

```
## early  late
##    26    61
```

```
## check if there are some NA to be removed:
which(is.na(GROUP))
```

```
## integer(0)
```

```
####################### DE ANALYSIS ############################

############################ edgeR
### common dispersion
d <- DGEList(counts = counts.f , group = GROUP)
names(d)
```

```
## [1] "counts"  "samples"
```

```
d <- estimateCommonDisp(d)
names(d)
```

```
## [1] "counts"            "samples"           "common.dispersion"
## [4] "pseudo.counts"     "pseudo.lib.size"   "AveLogCPM"
```

```
CD_parameter <- d$common.dispersion # 0.8715893
edgeR_common <- exactTest(d, pair=c("early", "late"), dispersion="common")

### tagwise dispersion
d <- estimateTagwiseDisp(d)
TGW <- data.frame(d$tagwise.dispersion)
colnames(TGW) <- "Tagwise dispersion"
head(TGW)
```

Tagwise dispersion
0.2379866
5.9306666
0.1020417
0.1545255
0.2207223
0.6277689

```
edgeR_TGW <- exactTest(d, pair=c("early", "late"), dispersion="tagwise")


#### get the values
topTags(edgeR_common) # first 10
```

```
## Comparison of groups:  late-early
##                         logFC     logCPM       PValue          FDR
## ENSG00000099399.5  -6.873890 -0.1283659 2.598395e-95 7.316560e-91
```

```
## ENSG00000205678.7  10.317190  1.9565364 1.076359e-78 1.515406e-74
## ENSG00000258444.1   9.475177 -0.9351235 1.792730e-70 1.682657e-66
## ENSG00000007350.16  8.408046  4.7564388 8.310980e-62 5.850515e-58
## ENSG00000092054.12  8.315099  3.6817563 5.615848e-61 3.162621e-57
## ENSG00000224127.1   8.061888 -2.8256951 2.278510e-57 1.059166e-53
## ENSG00000186910.3   7.958995 -1.5420518 2.633057e-57 1.059166e-53
## ENSG00000157005.3   7.894281 -0.3626381 5.131740e-57 1.806244e-53
## ENSG00000213934.6   7.746553  3.0783428 6.710277e-56 2.099422e-52
## ENSG00000198681.6   7.620094 -1.0292563 1.693400e-54 4.768276e-51
```

```
topTags(edgeR_TGW)
```

```
## Comparison of groups:  late-early
##                         logFC     logCPM       PValue          FDR
## ENSG00000185960.13 -4.636121 -1.3282189 3.132192e-18 8.819626e-14
## ENSG00000119715.14 -3.019493 -0.8021803 1.189827e-14 1.675158e-10
## ENSG00000229732.1  -3.967880  1.4491187 1.357988e-12 1.171090e-08
## ENSG00000149043.16 -3.529230  2.9839808 1.663599e-12 1.171090e-08
## ENSG00000171401.14 -3.554744  2.8139740 2.467385e-12 1.389532e-08
## ENSG00000196565.12  7.080971  3.9768259 3.505272e-10 1.645024e-06
## ENSG00000111339.10 -2.302114  0.1403517 5.610096e-10 2.071618e-06
## ENSG00000168356.11 -2.692913 -0.2955221 5.885696e-10 2.071618e-06
## ENSG00000064787.13  4.793299  1.0356488 8.759988e-10 2.740708e-06
## ENSG00000230257.1  -4.249184  0.1470924 1.204011e-09 3.390253e-06
```

```
sum(topTags(edgeR_common, n=Inf)$table$FDR < 0.05 ) # 2216 are significantlly DE
```

```
## [1] 2216
```

```
sum(topTags(edgeR_TGW, n=Inf)$table$FDR < 0.05 )  # 661
```

```
## [1] 661
```

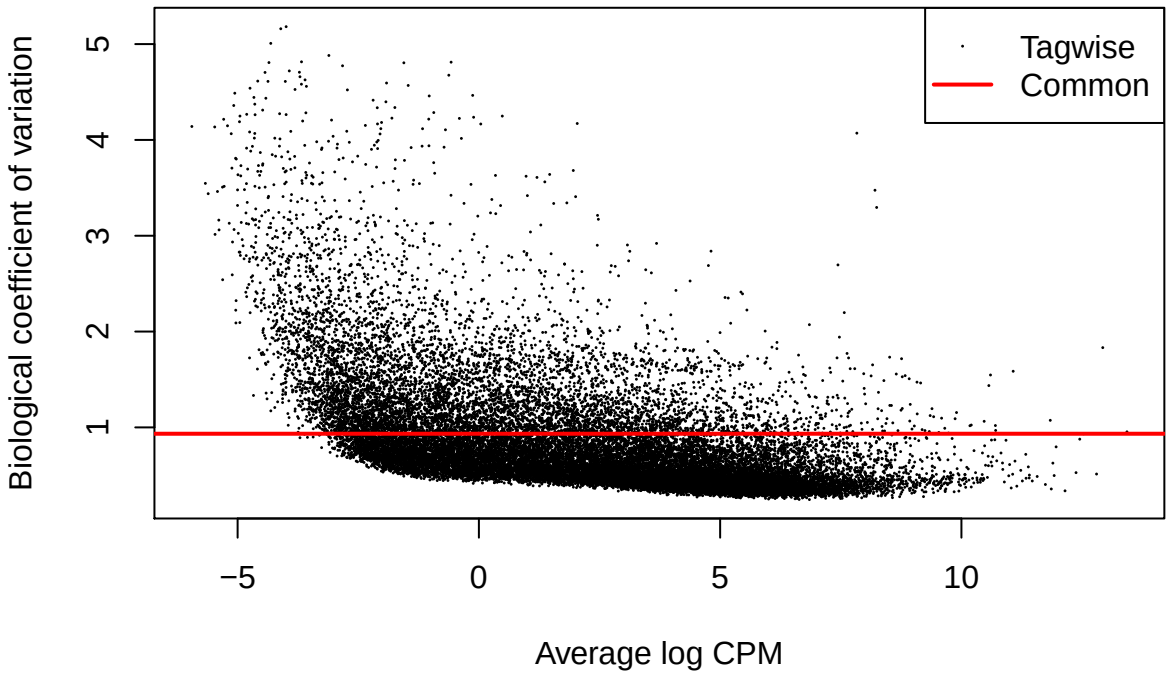
```
mean(topTags(edgeR_common, n=Inf)$table$FDR < 0.05) * 100 # 7.869877
```

```
## [1] 7.869877
```

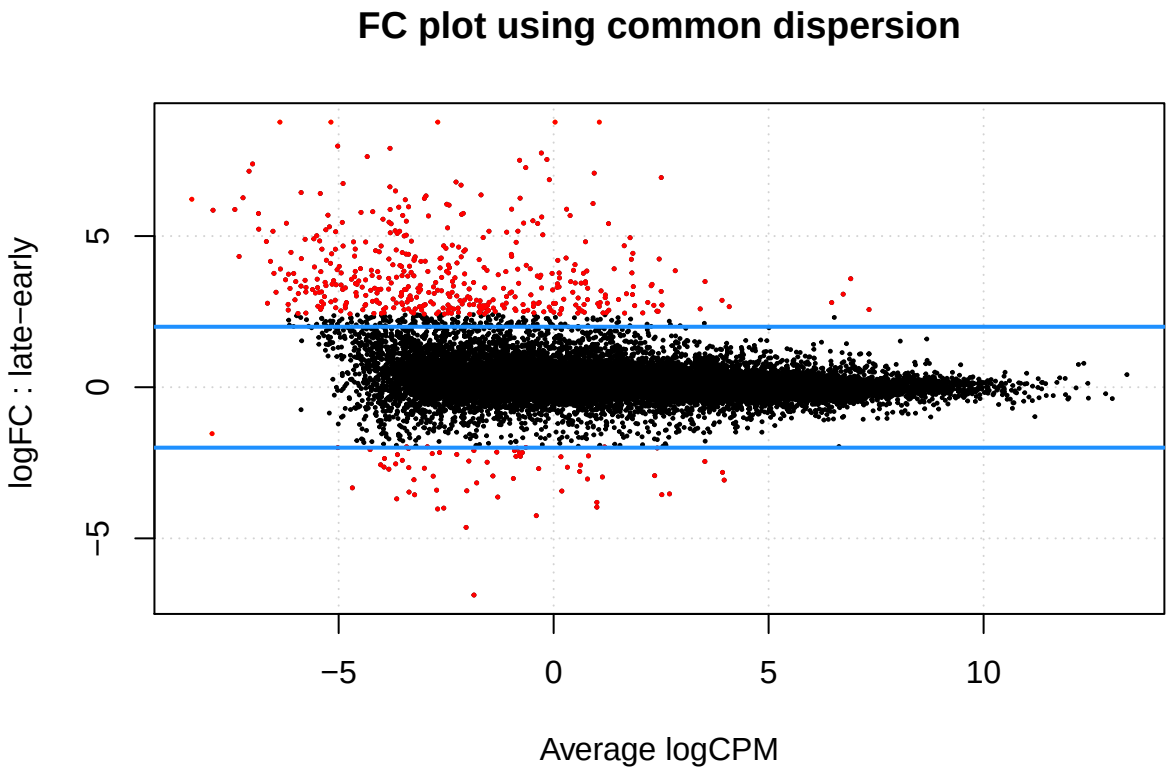
```
mean(topTags(edgeR_TGW, n=Inf)$table$FDR < 0.05) * 100  # 2.347468
```

```
## [1] 2.347468
```

```
plotBCV(d)
```




```
#### visualizing
detags500.com <- rownames(topTags(edgeR_common, n = 500)$table)
plotSmear(d, de.tags = detags500.com, main = "FC plot using common dispersion")
abline(h = c(-2, 2), col = "dodgerblue", lwd = 2)
```



```
####### selecting the genes with FDR < 0.05 and logFC > 1
DE_genes_CD <- topTags(edgeR_common, n=Inf)
mask <- DE_genes_CD$table$FDR < 0.05 &
        abs(DE_genes_CD$table$logFC) > log2(2)
DE_genes_selected_CD <-(DE_genes_CD$table[mask, ])
dim(DE_genes_selected_CD) # 2048
```

```
## [1] 2048    4
```

```
####### selecting the genes with FDR < 0.05 and logFC > 1 for tagwise dispersion:
DE_genes_TGW <- topTags(edgeR_TGW, n=Inf)
mask <- DE_genes_TGW$table$FDR < 0.05 &
        abs(DE_genes_TGW$table$logFC) > log2(2)
DE_genes_selected_TGW <-(DE_genes_TGW$table[mask, ])
dim(DE_genes_selected_TGW) # 620
```

```
## [1] 620   4
```

```
###### looking at up/down regulated genes:
DE_genes_selected_CD$updown <- factor(ifelse(DE_genes_selected_CD$logFC > 0, "up", "down"))
head(DE_genes_selected_CD)
```

	logFC	logCPM	PValue	FDR	updown
ENSG00000099399.5	-6.873890	-0.1283659	0	0	down
ENSG00000205678.7	10.317190	1.9565364	0	0	up
ENSG00000258444.1	9.475177	-0.9351235	0	0	up
ENSG00000007350.16	8.408046	4.7564388	0	0	up
ENSG00000092054.12	8.315099	3.6817563	0	0	up
ENSG00000224127.1	8.061888	-2.8256951	0	0	up

```
###### looking at up/down regulated genes with tagwise dispersion:
DE_genes_selected_TGW$updown <- factor(ifelse(DE_genes_selected_TGW$logFC > 0, "up", "down"))
head(DE_genes_selected_TGW)
```

	logFC	logCPM	PValue	FDR	updown
ENSG00000185960.13	-4.636121	-1.3282189	0	0.0e+00	down
ENSG00000119715.14	-3.019493	-0.8021803	0	0.0e+00	down
ENSG00000229732.1	-3.967880	1.4491187	0	0.0e+00	down
ENSG00000149043.16	-3.529230	2.9839808	0	0.0e+00	down
ENSG00000171401.14	-3.554744	2.8139740	0	0.0e+00	down
ENSG00000196565.12	7.080971	3.9768259	0	1.6e-06	up

```
#### storing all the genes in the variable geneUniverse:
geneUniverse <- (rownames(counts.f))
length(geneUniverse) # 28158
```

```
## [1] 28158
```

```
######################## ANNOTATION  ####################
library(clusterProfiler)
library(org.Hs.eg.db)

results_DE_genes_CD <- getBM(attributes = c("ensembl_gene_id_version", "entrezgene_id"), filters = "ens

length(which(!is.na(results_DE_genes_CD$entrezgene_id))) # 330
```

```
## [1] 330
```

```
results_DE_genes_TGW <- getBM(attributes = c("ensembl_gene_id_version", "entrezgene_id"), filters = "en

length(which(!is.na(results_DE_genes_TGW$entrezgene_id))) # 113
```

```
## [1] 113
```

```
geneUniverse <- getBM(attributes = c("ensembl_gene_id_version", "entrezgene_id"), filters = "ensembl_ge

length(which(!is.na(geneUniverse$entrezgene_id))) # 3938
```

```
## [1] 3938
```

```
######################## CLUSTER PROFILER ########################

enrich_go_bonferroni_CD <- enrichGO(gene          = results_DE_genes_CD$entrezgene_id,
                universe      = as.character(geneUniverse$entrezgene_id),
                OrgDb         = org.Hs.eg.db,
                ont           = "CC",
                pAdjustMethod = "bonferroni",
                pvalueCutoff  = 0.05,
                qvalueCutoff  = 0.05,
                readable = TRUE)

(summary(enrich_go_bonferroni_CD)) # 10 enriched terms found
```

	ID	Description	GeneRatio	BgRatio	pvalue	p.adjust
GO:0030017	GO:0030017	sarcomere	15/276	51/3484	4.90e-06	0.0014306
GO:0030016	GO:0030016	myofibril	15/276	55/3484	1.36e-05	0.0039882
GO:0099055	GO:0099055	integral component of postsynaptic membrane	10/276	27/3484	2.08e-05	0.0060869
GO:0044449	GO:0044449	contractile fiber part	15/276	57/3484	2.18e-05	0.0063901
GO:0031674	GO:0031674	I band	12/276	39/3484	2.69e-05	0.0078937
GO:0043292	GO:0043292	contractile fiber	15/276	59/3484	3.41e-05	0.0099906
GO:0099699	GO:0099699	integral component of synaptic membrane	11/276	34/3484	3.48e-05	0.0101847
GO:0098936	GO:0098936	intrinsic component of postsynaptic membrane	10/276	29/3484	4.27e-05	0.0125239
GO:0030018	GO:0030018	Z disc	11/276	36/3484	6.32e-05	0.0185224
GO:0099240	GO:0099240	intrinsic component of synaptic membrane	11/276	37/3484	8.37e-05	0.0245307

```
#### with tagwise dispersion:
enrich_go_bonferroni_TGW <- enrichGO(gene          = results_DE_genes_TGW$entrezgene_id,
                                     universe      = as.character(geneUniverse$entrezgene_id),
                                     OrgDb         = org.Hs.eg.db,
                                     ont           = "CC",
                                     pAdjustMethod = "bonferroni",
                                     pvalueCutoff  = 0.05,
                                     qvalueCutoff  = 0.05,
                                     readable = TRUE)

(summary(enrich_go_bonferroni_TGW)) # 8 enriched terms found
```

	ID	Description	GeneRatio	BgRatio	pvalue	p.adjust	qvalue	gene
GO:0030017	GO:0030017	sarcomere	9/91	51/3484	0.0000047	0.0010880	0.0007956	LDB
GO:0030016	GO:0030016	myofibril	9/91	55/3484	0.0000090	0.0020867	0.0007956	LDB
GO:0044449	GO:0044449	contractile fiber part	9/91	57/3484	0.0000122	0.0028283	0.0007956	LDB
GO:0043292	GO:0043292	contractile fiber	9/91	59/3484	0.0000163	0.0037842	0.0007984	LDB
GO:0031674	GO:0031674	I band	7/91	39/3484	0.0000512	0.0118767	0.0020046	LDB
GO:0099080	GO:0099080	supramolecular complex	15/91	197/3484	0.0001361	0.0315792	0.0033313	SCT
GO:0099081	GO:0099081	supramolecular polymer	15/91	197/3484	0.0001361	0.0315792	0.0033313	SCT
GO:0099512	GO:0099512	supramolecular fiber	15/91	197/3484	0.0001361	0.0315792	0.0033313	SCT

```
###### with FDR correction:
enrich_go_fdr_CD <- enrichGO(gene          = results_DE_genes_CD$entrezgene_id,
                             universe      = as.character(geneUniverse$entrezgene_id),
                             OrgDb         = org.Hs.eg.db,
                             ont           = "CC",
                             pAdjustMethod = "fdr",
                             pvalueCutoff  = 0.05,
                             qvalueCutoff  = 0.05,
                             readable = TRUE)

(summary(enrich_go_fdr_CD)) # 19 enriched terms found
```

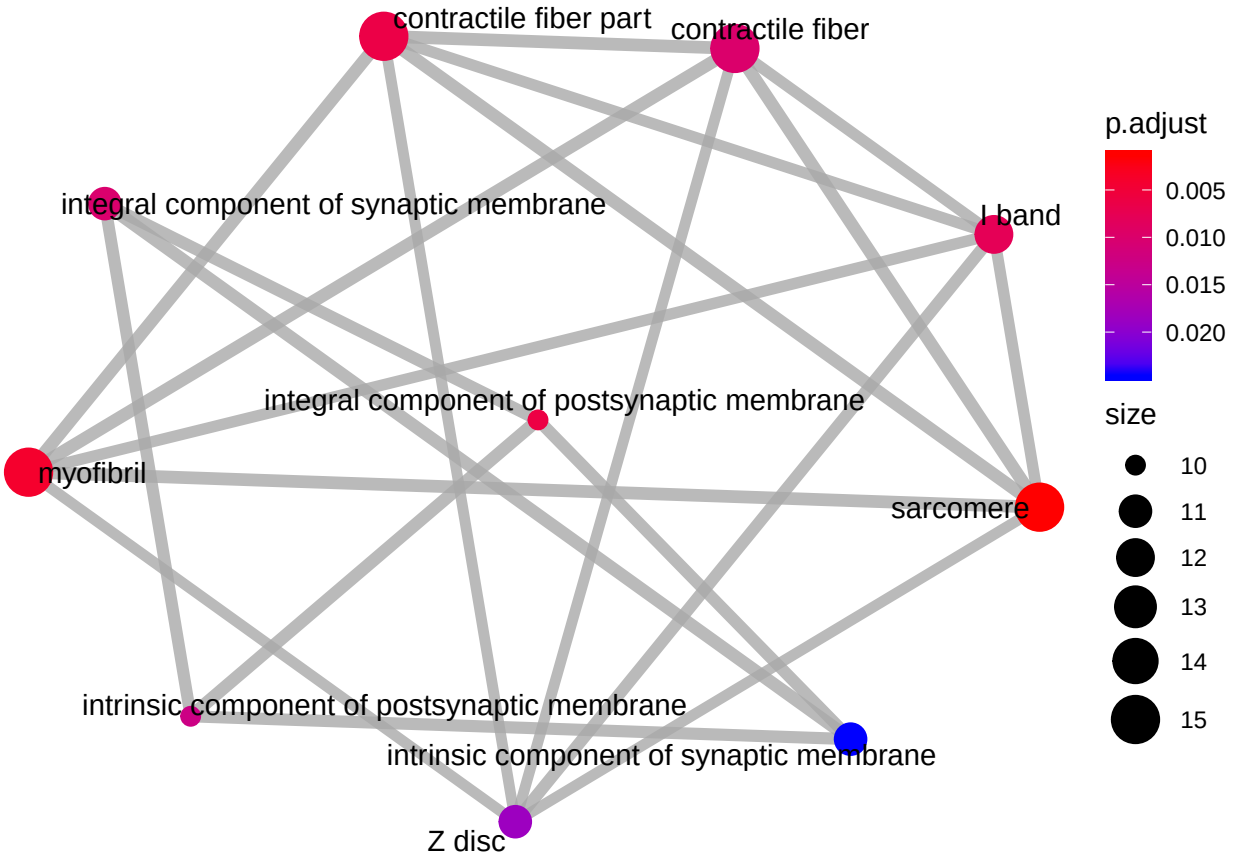
	ID	Description	GeneRatio	BgRatio	pvalue	p.adj
GO:0030017	GO:0030017	sarcomere	15/276	51/3484	0.0000049	0.0014
GO:0030016	GO:0030016	myofibril	15/276	55/3484	0.0000136	0.0014
GO:0099055	GO:0099055	integral component of postsynaptic membrane	10/276	27/3484	0.0000208	0.0014
GO:0044449	GO:0044449	contractile fiber part	15/276	57/3484	0.0000218	0.0014
GO:0031674	GO:0031674	I band	12/276	39/3484	0.0000269	0.0014
GO:0043292	GO:0043292	contractile fiber	15/276	59/3484	0.0000341	0.0014
GO:0099699	GO:0099699	integral component of synaptic membrane	11/276	34/3484	0.0000348	0.0014
GO:0098936	GO:0098936	intrinsic component of postsynaptic membrane	10/276	29/3484	0.0000427	0.0015
GO:0030018	GO:0030018	Z disc	11/276	36/3484	0.0000632	0.0020
GO:0099240	GO:0099240	intrinsic component of synaptic membrane	11/276	37/3484	0.0000837	0.0024
GO:0031226	GO:0031226	intrinsic component of plasma membrane	46/276	361/3484	0.0005237	0.0139
GO:1902495	GO:1902495	transmembrane transporter complex	14/276	68/3484	0.0006803	0.0153
GO:1990351	GO:1990351	transporter complex	14/276	68/3484	0.0006803	0.0153
GO:0097458	GO:0097458	neuron part	43/276	338/3484	0.0008437	0.0176
GO:0005887	GO:0005887	integral component of plasma membrane	44/276	353/3484	0.0011294	0.0220

	ID	Description	GeneRatio	BgRatio	pvalue	p.adj
GO:0045211	GO:0045211	postsynaptic membrane	14/276	72/3484	0.0012352	0.0226
GO:0034702	GO:0034702	ion channel complex	13/276	65/3484	0.0014004	0.0241
GO:0042383	GO:0042383	sarcolemma	8/276	30/3484	0.0017549	0.0285
GO:0034703	GO:0034703	cation channel complex	11/276	52/3484	0.0020210	0.0311

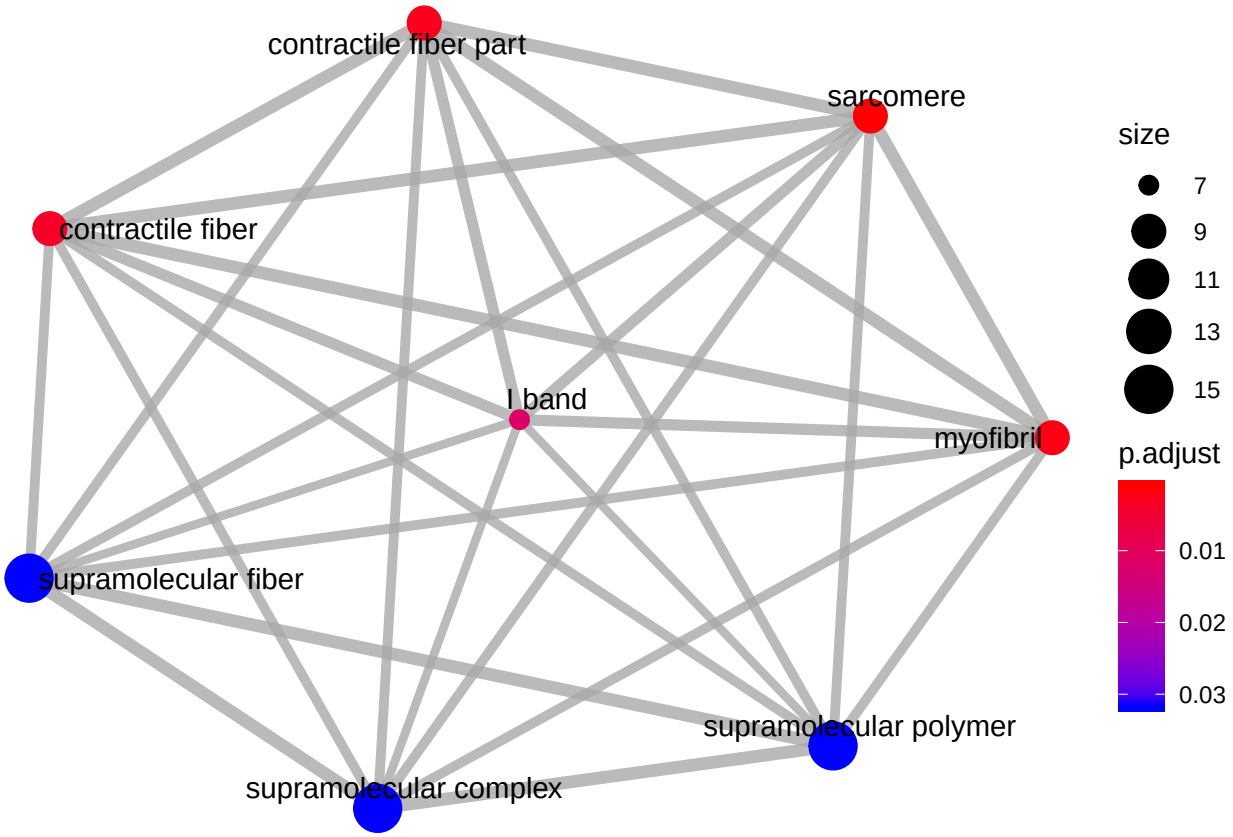
```
###### with tagwise dispersion:
enrich_go_fdr_TGW <- enrichGO(gene          = results_DE_genes_TGW$entrezgene_id,
                              universe      = as.character(geneUniverse$entrezgene_id),
                              OrgDb         = org.Hs.eg.db,
                              ont           = "CC",
                              pAdjustMethod = "fdr",
                              pvalueCutoff  = 0.05,
                              qvalueCutoff  = 0.05,
                              readable = TRUE)

(summary(enrich_go_fdr_TGW)) # 18 enriched terms found
```

	ID	Description	GeneRatio	BgRatio	pvalue	p.adjust	q
GO:0030017	GO:0030017	sarcomere	9/91	51/3484	0.0000047	0.0009428	0.000
GO:0030016	GO:0030016	myofibril	9/91	55/3484	0.0000090	0.0009428	0.000
GO:0044449	GO:0044449	contractile fiber part	9/91	57/3484	0.0000122	0.0009428	0.000
GO:0043292	GO:0043292	contractile fiber	9/91	59/3484	0.0000163	0.0009460	0.000
GO:0031674	GO:0031674	I band	7/91	39/3484	0.0000512	0.0023753	0.002
GO:0099080	GO:0099080	supramolecular complex	15/91	197/3484	0.0001361	0.0039474	0.003
GO:0099081	GO:0099081	supramolecular polymer	15/91	197/3484	0.0001361	0.0039474	0.003
GO:0099512	GO:0099512	supramolecular fiber	15/91	197/3484	0.0001361	0.0039474	0.003
GO:0030018	GO:0030018	Z disc	6/91	36/3484	0.0002787	0.0071848	0.006
GO:0000786	GO:0000786	nucleosome	6/91	50/3484	0.0016950	0.0284207	0.023
GO:0030424	GO:0030424	axon	10/91	129/3484	0.0017088	0.0284207	0.023
GO:1902495	GO:1902495	transmembrane transporter complex	7/91	68/3484	0.0017429	0.0284207	0.023
GO:1990351	GO:1990351	transporter complex	7/91	68/3484	0.0017429	0.0284207	0.023
GO:0044815	GO:0044815	DNA packaging complex	6/91	51/3484	0.0018812	0.0284207	0.023
GO:0042734	GO:0042734	presynaptic membrane	5/91	35/3484	0.0019022	0.0284207	0.023
GO:0032993	GO:0032993	protein-DNA complex	7/91	70/3484	0.0020658	0.0284207	0.023
GO:0034703	GO:0034703	cation channel complex	6/91	52/3484	0.0020826	0.0284207	0.023
GO:0005911	GO:0005911	cell-cell junction	8/91	95/3484	0.0030039	0.0387174	0.032

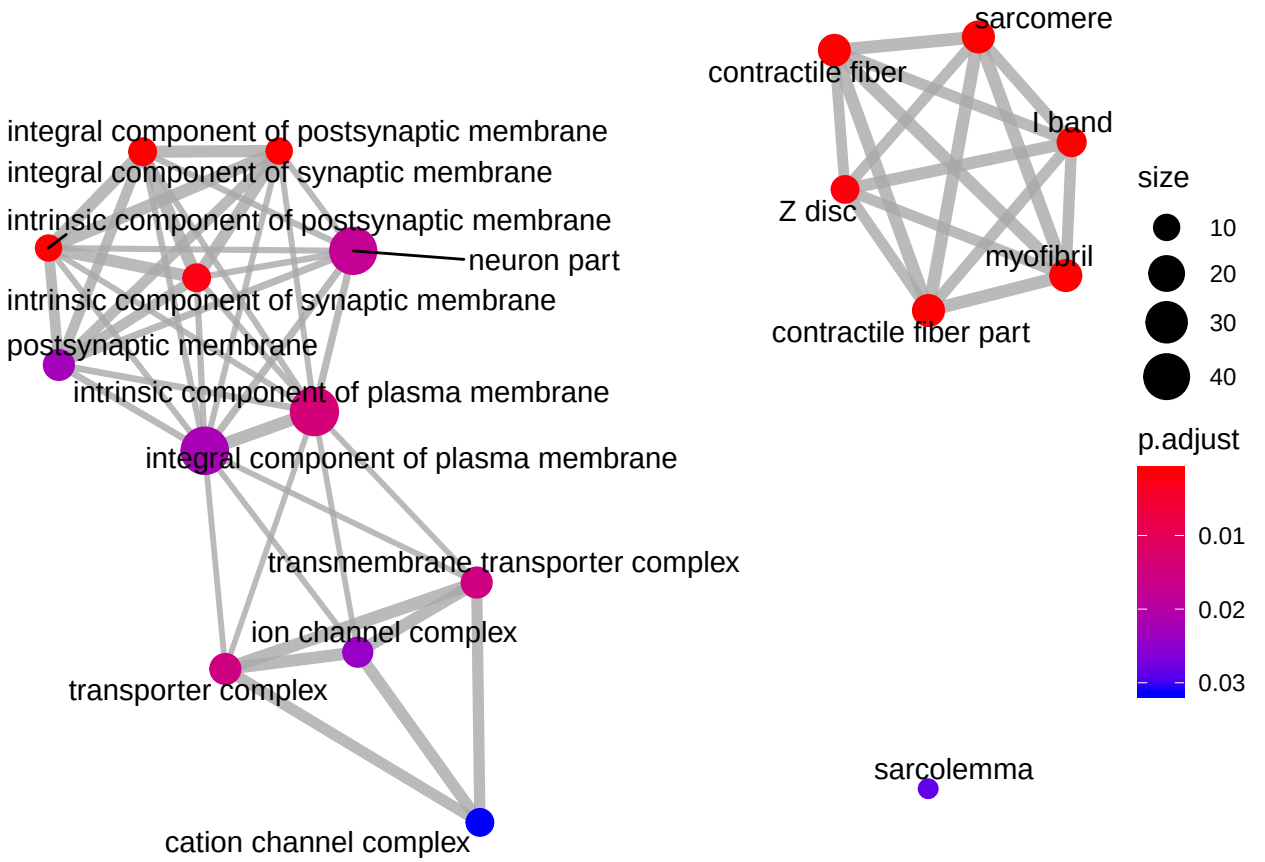
```
######################## VISUALIZATION ##############################
library(enrichplot)

plot_one_cd <- emapplot(enrich_go_bonferroni_CD)
plot_one_cd
```



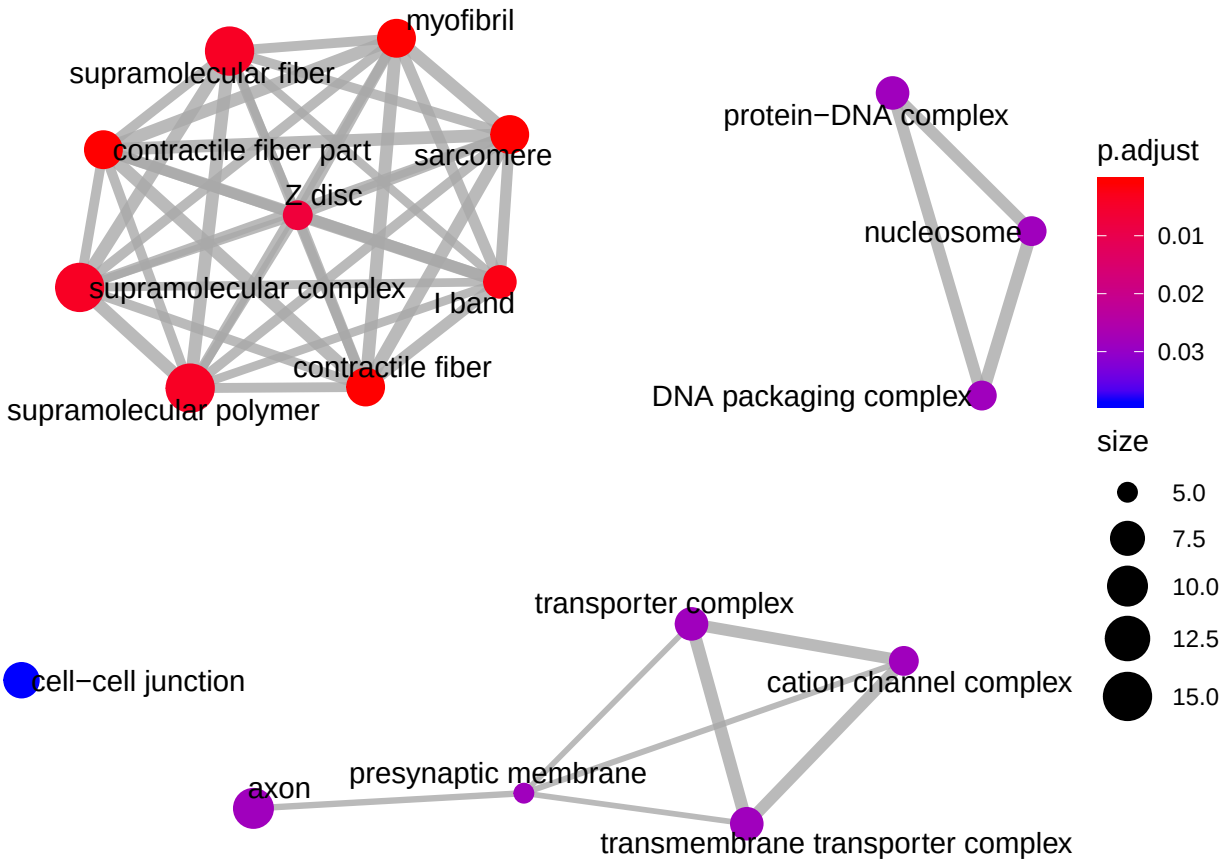

```
plot_one_tgw <- emapplot(enrich_go_bonferroni_TGW)
plot_one_tgw
```



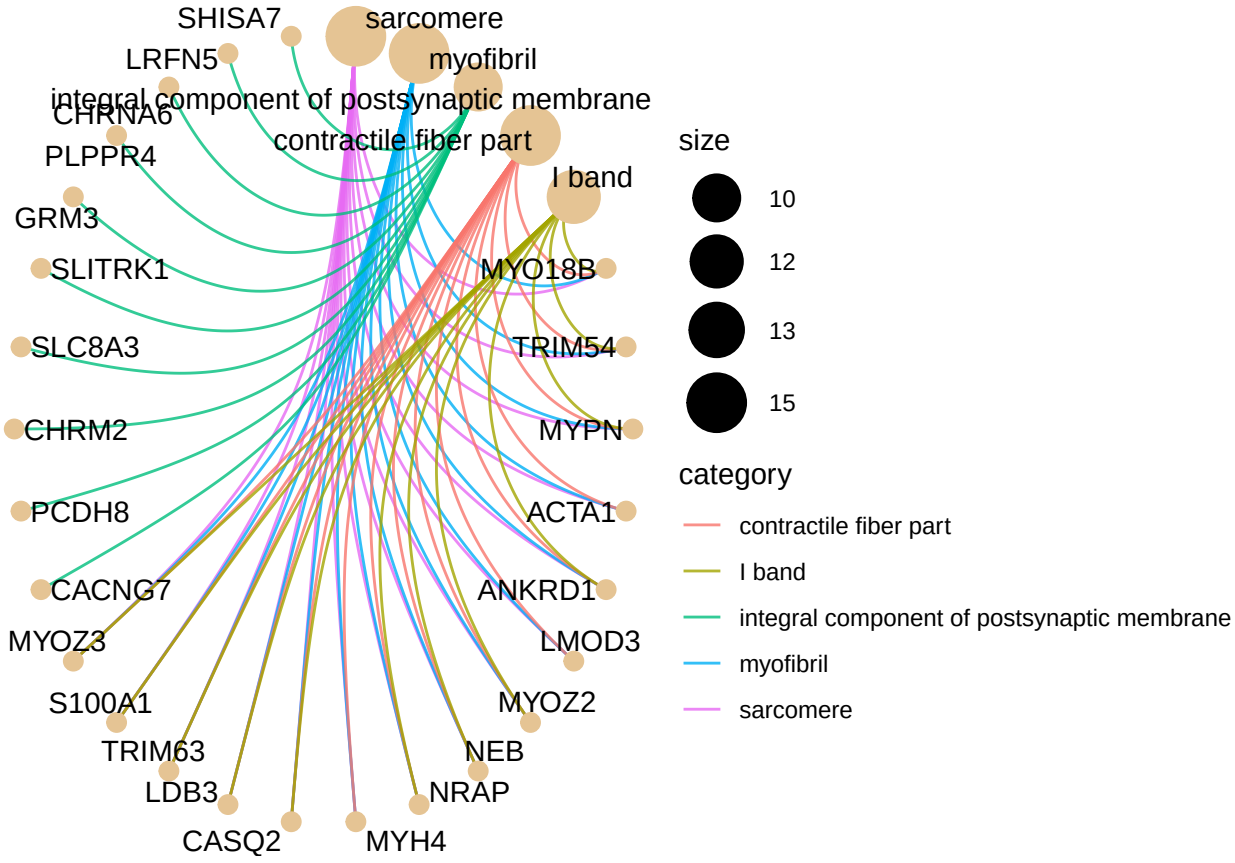

```
##############################################
plot_one_FDR_cd <- emapplot(enrich_go_fdr_CD)
plot_one_FDR_cd
```




```
plot_one_FDR_tgw <- emapplot(enrich_go_fdr_TGW)
plot_one_FDR_tgw
```




```
#### plot of best enrichment - bonferroni + CD
logFC_list <- (topTags(edgeR_common, n=Inf)$table$logFC)
circular_plot <- cnetplot(enrich_go_bonferroni_CD, foldChange=logFC_list, circular = TRUE, colorEdge = T
circular_plot
```


SHISA7
sarcomere
LRFN5
myofibril
integral component of postsynaptic membrane
CHRNA6
contractile fiber part
PLPPR4
I band
GRM3
SLITRK1
MYO18B
SLC8A3
TRIM54
CHRM2
MYPN
PCDH8
ACTA1
CACNG7
ANKRD1
MYOZ3
LMOD3
S100A1
MYOZ2
TRIM63
NEB
LDB3
NRAP
CASQ2
MYH4
size
10
12
13
15
category
contractile fiber part
I band
integral component of postsynaptic membrane
myofibril
sarcomere