A bio-functional polymer that prevents retinal scarring through modulation of NRF2 signalling pathway

Here we explore the use of DeSeq2 to extract differential expressed genes in epithelial cells treated under four conditions: Media only control, Polymer alone, TNF-α and TGF-β1 (TNT) alone, TNT + Polymer. The purpose of this pipeline is to replicate the results published by [Parikh et al., 2022](https://pubmed.ncbi.nlm.nih.gov/35589753/) which showed that upregulation of genes that are important for the NRF2 signalling pathway is associated with the presence of the polymer and not necessarily TNT.

library(GEOquery)

library(DESeq2)

library(dplyr)

library(tidyverse)

library(vsn)  
library(data.table)

# Set the working directory  
setwd("/Users/seymour/Desktop/Bioinformatics/Advanced\_Computational\_Biology\_BIOL0050/Week3/Gene\_Exp\_analysis\_3/")

gse <- getGEO(filename = "GSE176513\_series\_matrix.txt.gz", GSEMatrix = TRUE, getGPL = FALSE)

# Create a data frame with samples and conditions  
sample\_info <- pData(gse)  
condition\_df <- data.frame(Sample\_ID = sample\_info$description, Condition\_name = sample\_info$title) %>%  
 mutate(Sample\_ID = gsub("gene\_count\_matrix.csv --", "", Sample\_ID)) %>%  
 mutate(Condition = substr(Sample\_ID, 1, 1)) %>%  
 mutate(Duration = substr(Sample\_ID, 4, nchar(Sample\_ID))) %>%  
 mutate(cond\_duration = paste(Condition, Duration, sep = "\_")) %>%  
 mutate(Condition\_name = substr(Condition\_name, 1, nchar(Condition\_name) - 4)) %>%  
 mutate(Condition\_name\_short = gsub(" \\(8h\\)| \\(24h\\)", "", Condition\_name))

#Separate the 8h and 24h samples  
condition\_df\_8h <- condition\_df[condition\_df$Duration == "8h", ]  
condition\_df\_24h <- condition\_df[condition\_df$Duration == "24h", ]

#Read in the count data  
count\_data <- read.csv(gzfile("GSE176513\_gene\_count\_matrix.csv.gz"))

col\_order<- condition\_df$Sample\_ID  
rownames(count\_data) <- count\_data$EnsemblID  
aligned\_count\_data <- count\_data[, col\_order]

all\_dds <- DESeqDataSetFromMatrix(countData = aligned\_count\_data, colData = condition\_df, design = ~ Condition)

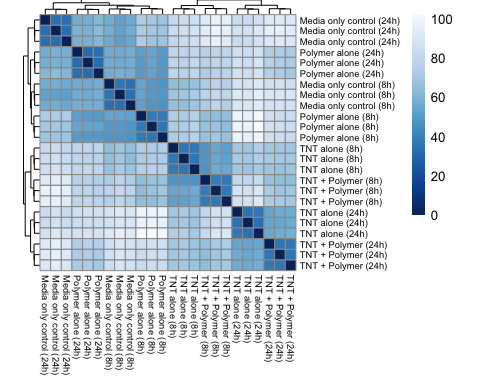
## Warning in DESeqDataSet(se, design = design, ignoreRank): some variables in  
## design formula are characters, converting to factors

all\_dds <- all\_dds[ rowSums(counts(all\_dds)) > 1, ]  
all\_dds <- estimateSizeFactors(all\_dds)

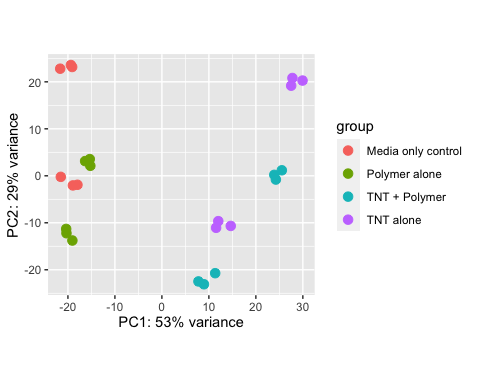
vsd <- vst(all\_dds, blind = FALSE)  
sampleDists\_vsd <- dist(t(assay(vsd)))  
sampleDistMatrix\_vsd <- as.matrix( sampleDists\_vsd )  
rownames(sampleDistMatrix\_vsd) <- paste(vsd$Condition\_name)  
colnames(sampleDistMatrix\_vsd) <- paste(vsd$Condition\_name)

### Distance matix of gene expression profiles from RNA sequencing of all four groups (Media only, 1 wt% poly(CEP), TNT and 1 wt% poly(CEP) + TNT at two time points (8 and 24 h), each done in triplicates. 24 h), each done in triplicates. The intensity of each box corresponds to the distance value shown in the legend.

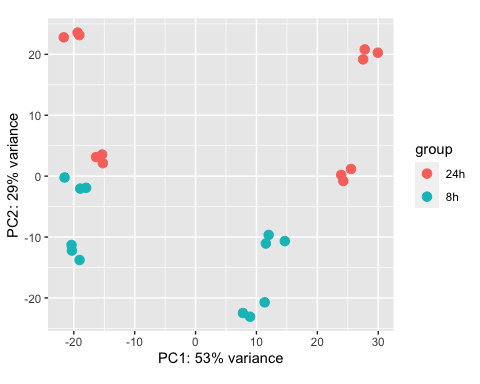
colors <- colorRampPalette(rev(brewer.pal(9, "Blues")))(255)  
heatmap <- pheatmap(sampleDistMatrix\_vsd,  
 clustering\_distance\_rows = sampleDists\_vsd,  
 clustering\_distance\_cols = sampleDists\_vsd,  
 col = colors,cellwidth=8,  
 cellheight=8,treeheight\_row = 8, treeheight\_col = 8, fontsize\_row = 7,fontsize\_col = 7)



pca\_data\_condition<-plotPCA(vsd, intgroup="Condition\_name\_short")  
pca\_data\_condition



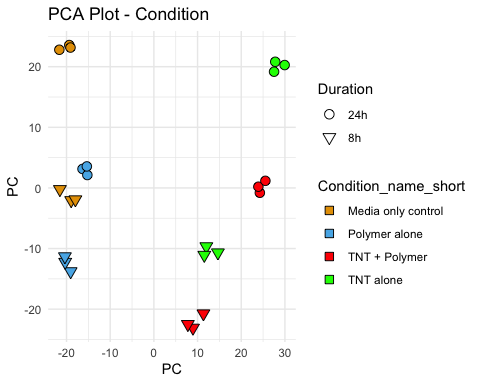
pca\_data\_duration<-plotPCA(vsd, intgroup="Duration")  
pca\_data\_duration



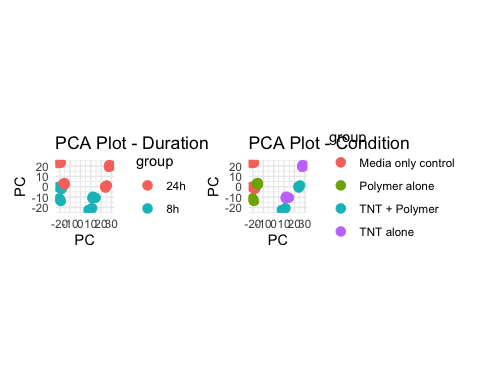
library(ggfortify)  
library(patchwork)  
  
pca\_data\_duration <- plotPCA(vsd, intgroup = "Duration")  
pca\_data\_condition <- plotPCA(vsd, intgroup = "Condition\_name\_short")  
match(pca\_data\_duration$data$name,pca\_data\_condition$data$name)

## [1] 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24

pca\_data\_condition$data$Duration<-pca\_data\_duration$data$Duration  
pcaData<-pca\_data\_condition$data  
ggplot(pcaData, aes(x= PC1, y = PC2))+geom\_point(size= 3, aes(shape=Duration, fill=Condition\_name\_short))+scale\_shape\_manual(values=c(21, 25))+ scale\_fill\_manual(values = c("#E69F00", "#56B4E9","Red","Green"),guide = guide\_legend(override.aes = list(shape = 22)))+scale\_color\_discrete()+labs(x = paste0("PC", pca\_data\_condition$x[, 1]), y = paste0("PC", pca\_data\_condition$x[, 2])) +ggtitle("PCA Plot - Condition") +theme\_minimal()



plot\_duration <- autoplot(pca\_data\_duration, data = vsd$Duration, shape = "Duration") +  
 labs(x = paste0("PC", pca\_data\_duration$x[, 1]), y = paste0("PC", pca\_data\_duration$x[, 2])) +  
 ggtitle("PCA Plot - Duration") +  
 theme\_minimal()  
  
# PCA plot with intgroup="Condition\_name\_short"  
pca\_data\_condition <- plotPCA(vsd, intgroup = "Condition\_name\_short")  
plot\_condition <- autoplot(pca\_data\_condition, data = vsd$Condition\_name\_short, colour = "Condition\_name\_short", shape="Duration") +  
 labs(x = paste0("PC", pca\_data\_condition$x[, 1]), y = paste0("PC", pca\_data\_condition$x[, 2])) +  
 ggtitle("PCA Plot - Condition") +  
 theme\_minimal()  
  
# Combine the two plots  
combined\_plot <- plot\_duration + plot\_condition  
  
# Display the combined plot  
print(combined\_plot)



#Create a function to investigate number of significant genes that are differentially expressed between conditions  
create\_dds <- function(condition\_xh) {  
 col\_order <- condition\_xh$Sample\_ID  
 aligned\_count\_data <- count\_data[, col\_order]  
 dds <- DESeqDataSetFromMatrix(countData = aligned\_count\_data, colData = condition\_xh, design = ~ Condition)  
 dds <- dds[ rowSums(counts(dds)) > 1, ]  
 dds <- estimateSizeFactors(dds)  
   
 dds <- DESeq(dds)  
   
 res<-results(dds, contrast=c("Condition","A","B"))  
 sigs<-na.omit(res)  
 sigs<-sigs[sigs$padj<0.05,]  
 cat("Number of genes sig in poly(CEP) alone compared to media only:", nrow(sigs), "\n")  
 #print(nrow(sigs))  
   
 res1<-results(dds, contrast=c("Condition","C","D"))  
 sigs1<-na.omit(res1)  
 sigs1<-sigs1[sigs1$padj<0.05,]  
 cat("Number of genes sig in poly(CEP)+TNT compared to TNT:", nrow(sigs1), "\n")  
   
 return(dds)  
}

dds\_8h <- create\_dds(condition\_df\_8h)

## Warning in DESeqDataSet(se, design = design, ignoreRank): some variables in  
## design formula are characters, converting to factors

## using pre-existing size factors

## estimating dispersions

## gene-wise dispersion estimates

## mean-dispersion relationship

## final dispersion estimates

## fitting model and testing

## Number of genes sig in poly(CEP) alone compared to media only: 3472   
## Number of genes sig in poly(CEP)+TNT compared to TNT: 3790

dds\_24h <- create\_dds(condition\_df\_24h)

## Warning in DESeqDataSet(se, design = design, ignoreRank): some variables in  
## design formula are characters, converting to factors

## using pre-existing size factors

## estimating dispersions

## gene-wise dispersion estimates

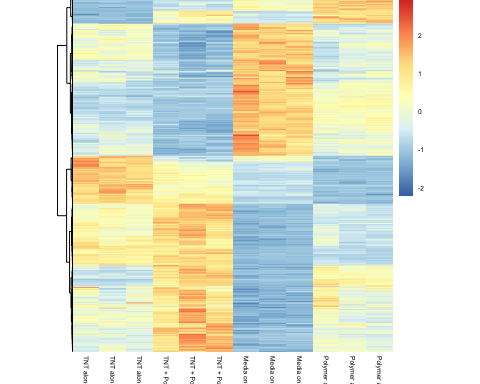
## mean-dispersion relationship

## final dispersion estimates

## fitting model and testing

## Number of genes sig in poly(CEP) alone compared to media only: 6646   
## Number of genes sig in poly(CEP)+TNT compared to TNT: 5162

res\_8h1<-results(dds\_8h,contrast=c("Condition","B","A"))  
res\_8h2<-results(dds\_8h,contrast=c("Condition","C","A"))  
res\_8h3<-results(dds\_8h,contrast=c("Condition","D","A"))  
  
create\_sigs\_df <- function(res\_xh) {  
 sigs\_xh <- na.omit(res\_xh)  
 sigs\_xh <- sigs\_xh[sigs\_xh$padj < 0.05,]  
 df\_sigs\_xh <- as.data.frame(sigs\_xh)  
 return(df\_sigs\_xh)  
}  
  
sigs\_8h1 <- create\_sigs\_df(res\_8h1)  
sigs\_8h2 <- create\_sigs\_df(res\_8h2)  
sigs\_8h3 <- create\_sigs\_df(res\_8h3)  
  
mat\_8h1<-counts(dds\_8h,normalized=T)[rownames(sigs\_8h1),]  
common\_row\_names1 <- intersect(rownames(mat\_8h1), rownames(sigs\_8h2))  
  
mat\_8h2<-mat\_8h1[common\_row\_names1,]  
common\_row\_names2 <- intersect(rownames(mat\_8h2), rownames(sigs\_8h3))  
  
mat\_8h3<-mat\_8h2[common\_row\_names2,]  
z\_mat\_8h3<-t(apply(mat\_8h3,1,scale))  
  
colnames(z\_mat\_8h3) <- condition\_df\_8h$Condition\_name\_short  
pheatmap(z\_mat\_8h3,clustering\_distance\_rows = "euclidean",clustering\_distance\_cols = "euclidean",clustering\_method = "ward.D2",treeheight\_row = 7,treeheight\_col = 20,show\_rownames = F, cellwidth=20, cellheight=0.2,fontsize=5)



# Your existing lists  
Media\_data <- rownames(sigs\_8h1)  
Polymer\_data <- rownames(sigs\_8h1)  
  
print(length(Media\_data))

## [1] 3472

TNT\_data <- rownames(sigs\_8h2)  
Polymer\_and\_TNT\_data <- rownames(sigs\_8h3)  
  
# Update Media with new rownames from sigs\_8h2 and sigs\_8h3  
Media\_data <- union(Media\_data, TNT\_data)  
print(length(Media\_data))

## [1] 7543

Media\_data <- union(Media\_data, Polymer\_and\_TNT\_data)  
print(length(Media\_data))

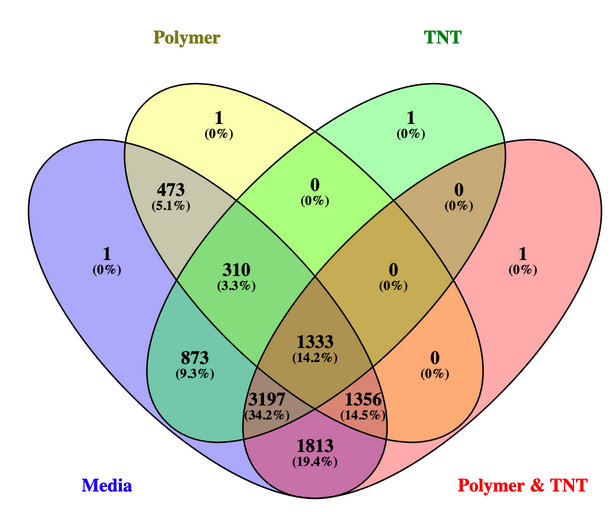
## [1] 9358

venn\_df\_Media <- data.frame(Media=Media\_data)  
venn\_df\_Polymer <- data.frame(Polymer=Polymer\_data)  
  
venn\_df\_Media$RowName <- venn\_df\_Media$Media  
venn\_df\_Polymer$RowName <-venn\_df\_Polymer$Polymer  
  
dat1 <- merge(venn\_df\_Media , venn\_df\_Polymer, by="RowName", all = T)  
  
  
venn\_df\_TNT <- data.frame(TNT=TNT\_data)  
venn\_df\_Polymer\_and\_TNT <- data.frame(Polymer\_and\_TNT=Polymer\_and\_TNT\_data)  
  
venn\_df\_TNT$RowName <- venn\_df\_TNT$TNT  
venn\_df\_Polymer\_and\_TNT$RowName <-venn\_df\_Polymer\_and\_TNT$Polymer\_and\_TNT  
  
dat2 <- merge(venn\_df\_TNT, venn\_df\_Polymer\_and\_TNT, by="RowName", all = T)  
dat3 <- merge(dat1, dat2, by="RowName", all = T)  
  
library(VennDiagram)

## Loading required package: futile.logger

#venn.diagram(dat3, filename = "venn-4-dimensions.png")  
  
#Error in `[[<-.data.frame`(`\*tmp\*`, i, value = c(NA, "ENSG00000000460", :  
#replacement has 3473 rows, data has 9358

write.csv(dat3, file = "dat3.csv", row.names = FALSE)

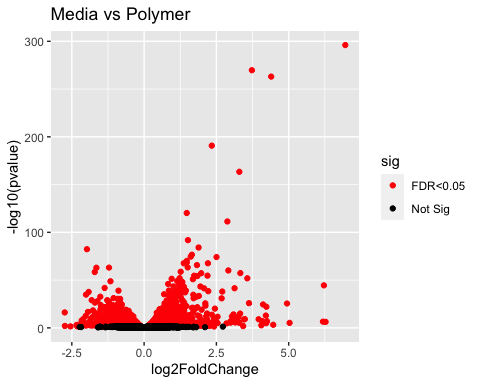


Fig

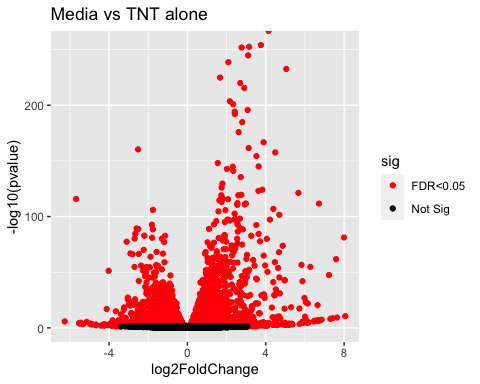
prepro\_res <- function(res\_xh) {  
 sigs\_xh <-res\_xh  
 sigs\_xh <- na.omit(res\_xh)  
 #sigs\_xh <- sigs\_xh[sigs\_xh$padj < 0.05,]  
 df\_sigs\_xh <- as.data.frame(sigs\_xh)  
 return(df\_sigs\_xh)  
}  
  
volc\_8h1 <- prepro\_res(res\_8h1)  
volc\_8h2 <- prepro\_res(res\_8h2)  
volc\_8h3 <- prepro\_res(res\_8h3)

#Fucntion to plot volcano plot  
make\_volc\_plot <- function(volc\_xhx,plot\_name) {  
 sigs\_xhx\_results <- volc\_xhx[order(volc\_xhx$pvalue),]  
 results\_order <- as.data.frame(dplyr::mutate(as.data.frame(sigs\_xhx\_results), sig=ifelse(sigs\_xhx\_results$pvalue<0.05, "FDR<0.05", "Not Sig")), row.names=rownames(sigs\_xhx\_results))  
   
 volcanoP <- ggplot(results\_order, ggplot2::aes(log2FoldChange, -log10(pvalue))) +  
 geom\_point(ggplot2::aes(col = sig)) +  
 scale\_color\_manual(values = c("red", "black")) +  
 ggtitle(plot\_name)  
   
 volcanoP  
}

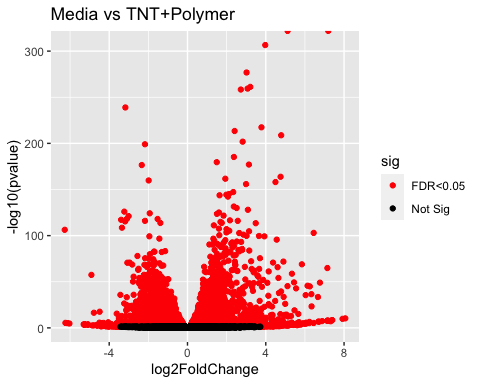
make\_volc\_plot(volc\_8h1,"Media vs Polymer")



make\_volc\_plot(volc\_8h2,"Media vs TNT alone")



make\_volc\_plot(volc\_8h3,"Media vs TNT+Polymer ")



library(org.Hs.eg.db)

## Loading required package: AnnotationDbi

##   
## Attaching package: 'AnnotationDbi'

## The following object is masked from 'package:dplyr':  
##   
## select

##

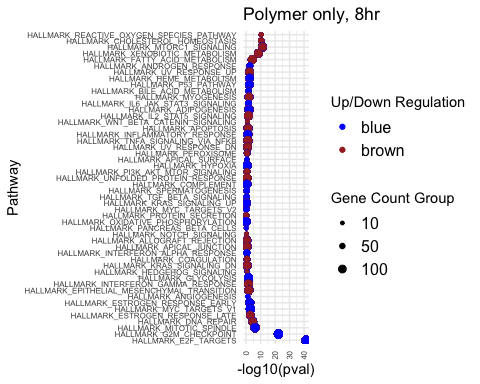
library(fgsea)  
  
pathways.hallmark <- gmtPathways("h.all.v7.2.symbols.gmt")  
  
plot\_pathways<-function(res\_xhx,plot\_title){  
   
 ens2symbol <- AnnotationDbi::select(org.Hs.eg.db,  
 key=rownames(res\_xhx),   
 columns="SYMBOL",  
 keytype="ENSEMBL")  
 ens2symbol <- as\_tibble(ens2symbol)  
   
 sigs\_xhx\_results <- res\_xhx[order(res\_xhx$pvalue),]  
 sigs\_xhx\_results$ENSEMBL <- rownames(sigs\_xhx\_results)  
 sigs\_xhx\_results <- inner\_join(sigs\_xhx\_results, ens2symbol, by=c("ENSEMBL"="ENSEMBL"))  
 sigs\_xhx\_results <- sigs\_xhx\_results %>%  
 mutate(updown = ifelse(log2FoldChange > 0, "upregulated", "downregulated"))  
   
 res2 <- sigs\_xhx\_results %>%   
 dplyr::select(SYMBOL, stat,updown) %>%   
 na.omit() %>%   
 distinct() %>%   
 group\_by(SYMBOL) %>%   
 summarize(stat=mean(stat), updown = first(updown))  
  
 ranks <- deframe(res2)  
   
 fgseaRes <- fgseaMultilevel(pathways=pathways.hallmark, stats=ranks)  
  
 # Tidy the results:  
 fgseaResTidy <- fgseaRes %>%  
 as\_tibble() %>%  
 arrange(desc(NES)) # order by normalized enrichment score (NES)  
   
 # To see what genes are in each of these pathways:  
 gene.in.pathway <- pathways.hallmark %>%   
 enframe("pathway", "SYMBOL") %>%   
 unnest(cols = c(SYMBOL)) %>%   
 inner\_join(res2, by="SYMBOL")  
   
 # Merge "updown" information with gene.in.pathway  
 gene.in.pathway <- gene.in.pathway %>%  
 inner\_join(res2, by = "SYMBOL")  
   
 # Merge the enriched pathways information with fgseaResTidy  
 fgseaResTidy <- fgseaResTidy %>%  
 inner\_join(gene.in.pathway, by = "pathway")  
   
 # Define the size breaks for the number of genes  
 size\_breaks <- c(10, 50, 100, 200,300, Inf)  
   
 # Calculate the circle size based on the number of genes in each pathway  
 fgseaResTidy$gene\_count\_group <- cut(fgseaResTidy$size, breaks = size\_breaks,   
 labels = c("10", "50", "100", "200","300"))  
   
 # Create a color palette for upregulation/downregulation  
 colors <- c("downregulated" = "blue", "upregulated" = "brown")  
   
 # Map colors to upregulation/downregulation  
 fgseaResTidy$circle\_color <- colors[fgseaResTidy$updown.x]  
   
 # Create the plot  
 ggplot(fgseaResTidy, aes(-log10(padj), reorder(pathway, NES))) +  
 geom\_point(aes(color = circle\_color, size = gene\_count\_group)) +  
 scale\_color\_manual(values = c("blue", "brown")) +  
 scale\_size\_manual(values = c("10" = 1, "50" = 1.6, "100" = 2.4, "200" = 5, "300" = 6)) +  
 theme\_minimal() +  
 theme(axis.text.x = element\_text(angle = 90, vjust = 0.5, hjust = 1, size = 6),  
 axis.text.y = element\_text(size = 6),  
 legend.text = element\_text(size = 12))+  
 labs(x = "-log10(pval)", y = "Pathway",   
 title = plot\_title) +  
 guides(size = guide\_legend(title = "Gene Count Group"),   
 color = guide\_legend(title = "Up/Down Regulation"))  
}

plot\_pathways(volc\_8h1,"Polymer only, 8hr")

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

## Warning: `x` must be a one- or two-column data frame in `deframe()`.

## Warning in preparePathwaysAndStats(pathways, stats, minSize, maxSize, gseaParam, : There are ties in the preranked stats (0.7% of the list).  
## The order of those tied genes will be arbitrary, which may produce unexpected results.

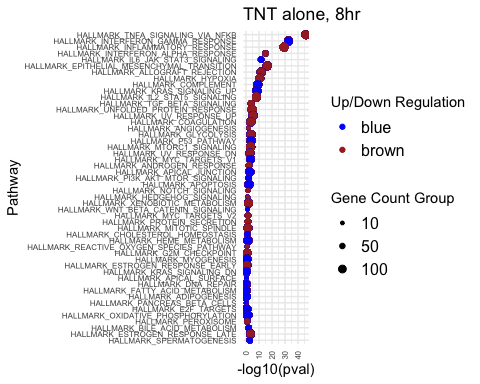


plot\_pathways(volc\_8h2,"TNT alone, 8hr")

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

## Warning: `x` must be a one- or two-column data frame in `deframe()`.

## Warning in preparePathwaysAndStats(pathways, stats, minSize, maxSize, gseaParam, : There are ties in the preranked stats (0.74% of the list).  
## The order of those tied genes will be arbitrary, which may produce unexpected results.



plot\_pathways(volc\_8h3,"TNT+ploymer, 8hr")

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

## Warning: `x` must be a one- or two-column data frame in `deframe()`.

## Warning in preparePathwaysAndStats(pathways, stats, minSize, maxSize, gseaParam, : There are ties in the preranked stats (0.8% of the list).  
## The order of those tied genes will be arbitrary, which may produce unexpected results.

