Steps:

1. Load read data and create DESeq2 objects for each organism/model type
2. Load read data and create DEXSeq objects for each organism/comparison/model type
3. Run DE/DEXSeq
4. Create PCAs and UpSet

```
################## Step 1 ##############################
# Please note that these object names may differ from the linux outputs
### First, DESeq2 for C elegans, then as linear, then G morhua, and linear again
### C elegans pairwise
setwd("~/Desktop/Honors/featureCounts/elegans/")
library(DESeq2)
library(tidyverse)

elegans <- read.table("counts_elegans", header=T)
elegans <- elegans[,-c(2:6)]
names(elegans) <- c("WBGENE","SRR11996348", "SRR11996349",
"SRR11996350","SRR11996351","SRR11996352","SRR11996353","SRR11996354","SRR11996
355","SRR11996356")
elegans$WBGENE <- str_remove(elegans$WBGENE, "^Gene:")
m <- as.matrix(elegans[,-1])
mode(m) <- "integer"
rownames(m) <- elegans$WBGENE
meta <- data.frame(colnames(m), Group = rep(c("Control","5ug_BaP","a20ug_BaP"), each=3))
#these are factors, numeric for an lm
dds <- DESeqDataSetFromMatrix(countData=m,
                              colData=meta,
                              design=~Group)

### C elegans Linear
meta_num <- data.frame(colnames(m), group = rep(c(0,5,20), each=3))
dds12 <- DESeqDataSetFromMatrix(countData=m,
                                colData=meta_num,
                                design=~group)

### G morhua pairwise
morhua <- read.table("counts_cod_v1", header=T)
morhua <- morhua[,-c(2:6)]
names(morhua) <-
c("Geneid","SRR6296791","SRR6296792","SRR6296793","SRR6296796","SRR6296797","SRR
6296798","SRR6296802","SRR6296803","SRR6296804","SRR6296808","SRR6296809","SRR6
296810",
"SRR6296815","SRR6296816","SRR6296817","SRR6296821","SRR6296822","SRR6296823","
SRR6296828","SRR6296829","SRR6296832","SRR6296833")
```

```
morhua$Geneid <- str_remove(morhua$Geneid, "^gene:")
mo <- as.matrix(morhua[,-1])
mode(mo) <- "integer"
rownames(mo) <- morhua$Geneid
meta2<- data.frame(colnames(mo),Group =
c("Control","0.01ug_BaP","1ug_BaP","Control","0.01ug_BaP","1ug_BaP","Control","0.01ug_BaP
","1ug_BaP","Control","0.01ug_BaP","1ug_BaP","Control","0.01ug_BaP","1ug_BaP","Control","0
.01ug_BaP","1ug_BaP","Control","1ug_BaP","Control","0.01ug_BaP"), ind =
c("a","a","a","b","b","b","c","c","c","d","d","d","e","e","e","f","f","f","g","g","h","h"))
dds2 <- DESeqDataSetFromMatrix(countData=mo,
                              colData=meta2,
                              design=~Group +ind)

### G morhua linear
meta2_num<- data.frame(colnames(mo), group =
c(0,0.01,1,0,0.01,1,0,0.01,1,0,0.01,1,0,0.01,1,0,0.01,1,0,1,0,0.01), ind =
c("a","a","a","b","b","b","c","c","c","d","d","d","e","e","e","f","f","f","g","g","h","h"))
dds2_num <- DESeqDataSetFromMatrix(countData=mo,
                         colData=meta2_num,
                         design=~ind+group)

################### Step 2 ###############################
library(DEXSeq)

### First, C elegans DEXSeq (5, 20uM, then linear), then G morhua (0.01, 1 uM, then linear) for
### Transcript level (DTU), then repeat for exon level (DEU)
### C elegans 5 uM DTU
dexelegans <- read.table("dexseq_reads_elegans", header=T)
dexelegans <- dexelegans[,-c(2:6)]
names(dexelegans) <- c("WBGENE","SRR11996348", "SRR11996349",
"SRR11996350","SRR11996351","SRR11996352","SRR11996353","SRR11996354","SRR11996
355","SRR11996356")

dexm <- as.matrix(dexelegans[,-1])
mode(dexm) <- "integer"
rownames(dexm) <- dexelegans$WBGENE

sample.data.elegans <- read.table("transcript2gene.txt")
names(sample.data.elegans) <- c("WBGene","TranscriptID")
sample.data.elegans$TranscriptID <- str_remove(sample.data.elegans$TranscriptID, "^Name=")
sample.data.elegans$WBGene <- str_remove(sample.data.elegans$WBGene,
"^Parent=Gene:")

dexm_16 <- dexm[,c(1:6)]
```

```
meta_16 <- meta[c(1:6),]
dxd05 <- DEXSeqDataSet(countData = dexm_16, #featureCounts output
              sampleData = meta_16, #same as de
              design=~sample + exon + group:exon, #but it's actually transcript
              featureID = sample.data.elegans$TranscriptID, #the transcripts
              groupID = sample.data.elegans$WBGene) #genes they belong to

### C elegans 20 uM DTU
dexm_020 <- dexm[,c(1:3, 7:9)]
meta_020 <- meta[c(1:3, 7:9),]
dxd020 <- DEXSeqDataSet(countData = dexm_020, #featureCounts output
               sampleData = meta_020, #same as de
               design=~sample + exon + group:exon, #but it's actually transcript
               featureID = sample.data.elegans$TranscriptID, #the transcripts
               groupID = sample.data.elegans$WBGene) #genes they belong to

### C elegans linear DTU
meta_num <- data.frame(colnames(m), group = rep(c(0,5,20), each=3))
dxd_num <- DEXSeqDataSet(countData=dexm,
                     sampleData=meta_num,
                     design=~sample + exon + group:exon,
                     featureID = sample.data.elegans$TranscriptID,
                     groupID = sample.data.elegans$WBGene)

### C elegans 5 uM DEU
exonelegans <- read.table("exoncountselegans2.txt", header=T)
exonelegans <- exonelegans[,-c(2:6)]
names(exonelegans) <- c("GeneID","SRR11996348", "SRR11996349",
"SRR11996350","SRR11996351","SRR11996352","SRR11996353","SRR11996354","SRR11996
355","SRR11996356")

exonm <- as.matrix(exonelegans[,-1])
mode(exonm) <- "integer"
rownames(exonm) <- exonelegans$GeneID
exons <- as.data.frame(exonm)
exons$WBGene <- gsub("\\..*$", "", rownames(exons))
exons$exonname <- rownames(exons)

exonm05 <- exonm[,c(1:6)]
# meta_16
dxdexon05 <- DEXSeqDataSet(countData = exonm05, #featureCounts output
              sampleData = meta_16, #same as de
              design=~sample + exon + Group:exon,
              featureID = exons$exonname, #the exons
```

```
                groupID = exons$WBGene) #genes they belong to

### C elegans 20 uM DEU
exonm020 <- exonm[,c(1:3,7:9)]
# meta_020 is conserved from DTU object
dxdexon020 <- DEXSeqDataSet(countData = exonm020, #featureCounts output
                  sampleData = meta_020, #same as de
                  design=~sample + exon + group:exon,
                  featureID = exons$exonname, #the exons
                  groupID = exons$WBGene) #genes they belong to

### C elegans linear DEU
ddsexonnum <- DEXSeqDataSet(countData=exonm,
                        sampleData = meta_num,
                        design=~sample + exon + group:exon,
                        featureID = exons$exonname,
                        groupID = exons$WBGene)

### G morhua 0.01 uM DTU
dexmorhua <- read.table("dexseq_counts_cod_v1", header=T)
dexmorhua <- dexmorhua[,-c(2:6)]
names(dexmorhua) <-
c("TranscriptID","SRR6296791","SRR6296792","SRR6296793","SRR6296796","SRR6296797","
SRR6296798","SRR6296802","SRR6296803","SRR6296804","SRR6296808","SRR6296809","
SRR6296810",
"SRR6296815","SRR6296816","SRR6296817","SRR6296821","SRR6296822","SRR6296823","
SRR6296828","SRR6296829","SRR6296832","SRR6296833")
dexmorhua$TranscriptID <- str_remove(dexmorhua$TranscriptID, "^transcript:")
dexmorhua$Transcript <- dexmorhua$TranscriptID
sample.data.morhua <- read.table("morhua_transcript_metadata.tsv")
names(sample.data.morhua) <- c("TranscriptID","GeneID")
sample.data.morhua$TranscriptID <- str_remove(sample.data.morhua$TranscriptID,
"^transcript:")
sample.data.morhua$GeneID <- str_remove(sample.data.morhua$GeneID, "^gene:")
sample.data.morhua$TranscriptID <- str_remove(sample.data.morhua$TranscriptID, "^*;")
sample.data.morhua$GeneID <- str_remove(sample.data.morhua$GeneID, "^*;")
dexmorhua <- merge(dexmorhua, sample.data.morhua, by.all = TranscriptID)
Transcript <- dexmorhua$TranscriptID
Gene <- dexmorhua$GeneID
dexmo <- as.matrix(dexmorhua[,-c(1,24,25)])
mode(dexmo) <- "integer"
rownames(dexmo) <- dexmorhua$TranscriptID
```

```
dxdmo0.01 <- DEXSeqDataSet(countData = dexmo[,c(1,2,4,5,7,8,10,11,13,14,16,17,19,21,22)],
#featureCounts output
                 sampleData = meta2[c(1,2,4,5,7,8,10,11,13,14,16,17,19,21,22),-3], #same as de
                 design=~sample + exon + group:exon, #but it's actually transcript
                 featureID = Transcript, #the transcripts
                 groupID = Gene) #genes they belong to

### G morhua 1 uM DTU
dxdmo1 <- DEXSeqDataSet(countData = dexmo[,c(1,3,4,6,7,9,10,12,13,15,16,18,19,20,21)],
#featureCounts output
                      sampleData = meta2[c(1,3,4,6,7,9,10,12,13,15,16,18,19,20,21),], #same as de
                      design=~sample + exon + group:exon, #but it's actually transcript
                      featureID = Transcript, #the transcripts
                      groupID = Gene) #genes they belong to

### G morhua linear DTU
meta2_num_dex <- meta2_num[,c(1,2)]
dex_num_mo <- DEXSeqDataSet(countData=dexmo,
                    sampleData=meta2_num_dex,
                    design=~sample+exon+group:exon,
                    featureID = Transcript,
                    groupID = Gene)

### G morhua 0.01 uM DEU
exonmorhua <- read.table("exoncountsmorhua.txt", header = T)
exonmorhua <- exonmorhua[,-c(2:6)]
names(exonmorhua) <-
c("Geneid","SRR6296791","SRR6296792","SRR6296793","SRR6296796","SRR6296797","SRR
6296798","SRR6296802","SRR6296803","SRR6296804","SRR6296808","SRR6296809","SRR6
296810",
"SRR6296815","SRR6296816","SRR6296817","SRR6296821","SRR6296822","SRR6296823","
SRR6296828","SRR6296829","SRR6296832","SRR6296833")
exonmo <- as.matrix(exonmorhua[,-1])
mode(exonmo) <- "integer"
rownames(exonmo) <- exonmorhua$Geneid

exonmo <- as.data.frame(exonmo)
exonmo$GeneID <- gsub("\\..*$", "", rownames(exonmo))
exonmo$exonname <- rownames(exonmo)
moexonname <- exonmo$exonname
moexongene <- exonmo$GeneID
exonmo <- exonmo[,-c(23,24)]
```

```
dxexonmo0.01 <- DEXSeqDataSet(countData =
exonmo[,c(1,2,4,5,7,8,10,11,13,14,16,17,19,21,22)], #featureCounts output
                        sampleData = meta2[c(1,2,4,5,7,8,10,11,13,14,16,17,19,21,22),], #same as de
                        design=~sample + exon + group:exon,
                        featureID = moexonname, #the exons
                        groupID = moexongene) #genes they belong to

### G morhua 1 uM DEU
dxexonmo1 <- DEXSeqDataSet(countData =
exonmo[,c(1,3,4,6,7,9,10,12,13,15,16,18,19,20,21)], #featureCounts output
                           sampleData = meta2[c(1,3,4,6,7,9,10,12,13,15,16,18,19,20,21),], #same as
de
                           design=~sample + exon + group:exon,
                           featureID = moexonname, #the exons
                           groupID = moexongene) #genes they belong to

### G morhua linear DEU
dexex_num_mo <- DEXSeqDataSet(countData=exonmo,
                          sampleData=meta2_num_dex,
                          design=~sample+exon+group:exon,
                          featureID = moexonname,
                          groupID = moexongene)

################### Step 3 ###############################
### Code is described in the same order as objects were created above
### DESeq2 for both organisms, then DTU/DEU for both organisms
### C elegans pairwise DGE - we subset the working data frame first
counts(dds)
smallestGroupSize <- 3
keep <- rowSums(counts(dds) >= 10) >= smallestGroupSize
ddselegans1 <- dds[keep,]
elegansde1 <- DESeq(ddselegans1) #object used for results
res5 <- results(elegansde1, alpha = 0.05, lfcThreshold = 1, contrast =
c("group","5ug_BaP","Control"))
res20 <- results(elegansde1, alpha = 0.05, lfcThreshold = 1, contrast =
c("group","20ug_BaP","Control"))

summary(res5)
res5df <- as.data.frame(res5)
res5df1 <- subset(res5df, log2FoldChange > 1 & padj < 0.05) #just to remember where DEGs
came from
res20df <- as.data.frame(res20)
res20df1 <- subset(res20df, log2FoldChange > 1 & padj < 0.05)
res20df2 <- subset(res20df, log2FoldChange < -1 & padj < 0.05)
```

```
res20df3 <- rbind(res20df1, res20df2)

### C elegans linear DGE
smallestGroupSizeelenum <- 3
keepelenum <- rowSums(counts(dds12) >= 10) >= smallestGroupSizeelenum
ddselegansnum <- dds12[keepelenum,]
elegansde2 <- DESeq(ddselegansnum)
resnum <- results(elegansde2)
resnumalpha <- subset(resnum, padj < 0.05)
resnumalphadf <- as.data.frame(resnumalpha)
resnumalphadfsub <- subset(resnumalphadf, log2FoldChange > 0) #this info not used in paper
resnumalphadfsubneg <- subset(resnumalphadf, log2FoldChange < 0) #same

### G morhua pairwise DGE
smallestGroupSizeMo <- 3
keep <- rowSums(counts(dds2) >= 10) >= smallestGroupSizeMo
dds2 <- dds2[keep,]
morhuade1 <- DESeq(dds2) #object stores model
res0.01 <- results(morhuade1, contrast = c("Group", "0.01ug_BaP","Control"), alpha = 0.05,
lfcThreshold = 1)
res1 <- results(morhuade1, contrast = c("Group","1ug_BaP","Control"), alpha = 0.05,
lfcThreshold = 1)
res0.01df <- as.data.frame(res0.01)
res0.01df <- subset(res0.01df, log2FoldChange > 1 & padj < 0.05)
res1df <- as.data.frame(res1)
res1df1 <- subset(res1df, log2FoldChange > 1 & padj < 0.05)

### G morhua linear DGE - object not subset-able because of design in linear model
morhuade_num <- DESeq(dds2_num)
morhuade_numdf <- results(morhuade_num)
morhuade_numdf <- as.data.frame(morhuade_numdf)
morhuade_num_sig <- subset(morhuade_numdf, padj < 0.05)

### C elegans 5 uM DTU
system.time({
  dxd05 <- estimateSizeFactors(dxd05)
  dxd05 <- estimateDispersions(dxd05, quiet = T)
  dxd05 <- testForDEU(dxd05, reducedModel =~sample + exon) # also try coding as a number,
getting a fc/unit
}) # stratify model, only including 1 treatment and control

dx05red <- DEXSeqResults(dxd05, independentFiltering = F)
qval05 <- perGeneQValue(dx05red)
dxr.g05 <- data.frame(gene=names(qval05),qval05)
```

```
subset(dxr.g05, qval05 < 0.1)

### C elegans 20 uM DTU
system.time({
  dxd020 <- estimateSizeFactors(dxd020)
  dxd020 <- estimateDispersions(dxd020, quiet = T)
  dxd020 <- testForDEU(dxd020, reducedModel =~sample + exon) # also try coding as a
number, getting a fc/unit
}) # stratify model, only including 1 treatment and control

dx020red <- DEXSeqResults(dxd020, independentFiltering = F)
qval020 <- perGeneQValue(dx020red)
dxr.g020 <- data.frame(gene=names(qval020),qval020)
subset(dxr.g020, qval020 < 0.1)

### C elegans linear DTU
system.time({
  dxd_num <- estimateSizeFactors(dxd_num)
  dxd_num <- estimateDispersions(dxd_num, quiet = T)
  dxd_num <- testForDEU(dxd_num, reducedModel =~sample + exon) # also try coding as a
number, getting a fc/unit
}) # stratify model, only including 1 treatment and control

dx_numred <- DEXSeqResults(dxd_num, independentFiltering = F)
qvalnum <- perGeneQValue(dx_numred)
dxr.gnum <- data.frame(gene=names(qvalnum),qvalnum)
dxr.gnum_sig <- subset(dxr.gnum, qvalnum < 0.05)

### C elegans 5 uM DEU
system.time({
  dxdexon05 <- estimateSizeFactors(dxdexon05)
  dxdexon05 <- estimateDispersions(dxdexon05, quiet = T)
  dxdexon05 <- testForDEU(dxdexon05, reducedModel =~sample + exon)
})

dxexonred05 <- DEXSeqResults(dxdexon05, independentFiltering = F)

qvalex05 <- perGeneQValue(dxexonred05)
dxr.gex05 <- data.frame(gene=names(qvalex05),qvalex05)
dxrgex05sig <- subset(dxr.gex05, qvalex05 < 0.05) #19 sig genes

### C elegans 20 uM DEU
system.time({
  dxdexon020 <- estimateSizeFactors(dxdexon020)
  dxdexon020 <- estimateDispersions(dxdexon020, quiet = T)
  dxdexon020 <- testForDEU(dxdexon020, reducedModel =~sample + exon)
})

dxexonred020 <- DEXSeqResults(dxdexon020, independentFiltering = F)

qvalex020 <- perGeneQValue(dxexonred020)
dxr.gex020 <- data.frame(gene=names(qvalex020),qvalex020)
dxrgex020sig <- subset(dxr.gex020, qvalex020 < 0.05) #29 sig genes

### C elegans linear DEU
system.time({
  ddsexonnum <- estimateSizeFactors(ddsexonnum)
  ddsexonnum <- estimateDispersions(ddsexonnum, quiet = T)
  ddsexonnum <- testForDEU(ddsexonnum, reducedModel =~sample + exon)
})
ddsexonnumres <- DEXSeqResults(ddsexonnum, independentFiltering = F)

qvalexnum <- perGeneQValue(ddsexonnumres)
dxr.gexnum <- data.frame(gene=names(qvalexnum),qvalexnum)
dxrgexnumsig <- subset(dxr.gexnum, qvalexnum < 0.05)

### G morhua 0.01 uM DTU
system.time({
  dxdmo0.01 <- estimateSizeFactors(dxdmo0.01)
  dxdmo0.01 <- estimateDispersions(dxdmo0.01, quiet = T)
  dxdmo0.01 <- testForDEU(dxdmo0.01, reducedModel =~sample + exon)
})

dxredmo0.01 <- DEXSeqResults(dxdmo0.01, independentFiltering = F)

qval_mor_dex0.01 <- perGeneQValue(dxredmo0.01)
dxr.g_mor_dex0.01 <- data.frame(gene=names(qval_mor_dex0.01),qval_mor_dex0.01)
subset(dxr.g_mor_dex0.01, qval_mor_dex0.01 < 0.1)

### G morhua 1 uM DTU
system.time({
  dxdmo1 <- estimateSizeFactors(dxdmo1)
  dxdmo1 <- estimateDispersions(dxdmo1, quiet = T)
  dxdmo1 <- testForDEU(dxdmo1, reducedModel =~sample + exon)
})

dxredmo1 <- DEXSeqResults(dxdmo1, independentFiltering = F)
```

```
qval_mor_dex1 <- perGeneQValue(dxredmo1)
dxr.g_mor_dex1 <- data.frame(gene=names(qval_mor_dex1),qval_mor_dex1)
subset(dxr.g_mor_dex1, qval_mor_dex1 < 0.1)

### G morhua linear DTU
system.time({
  dex_num_mo <- estimateSizeFactors(dex_num_mo)
  dex_num_mo <- estimateDispersions(dex_num_mo, quiet = T)
  dex_num_mo <- testForDEU(dex_num_mo, reducedModel =~sample + exon)
})
dex_num_mo_res <- DEXSeqResults(dex_num_mo, independentFiltering = F)
qval_mo_dex <- perGeneQValue(dex_num_mo_res)
dxr.g_mo_dex <- data.frame(gene=names(qval_mo_dex),qval_mo_dex)
subset(dxr.g_mo_dex, qval_mo_dex < 0.05)

### G morhua 0.01 uM DEU
system.time({
  dxexonmo0.01 <- estimateSizeFactors(dxexonmo0.01)
  dxexonmo0.01 <- estimateDispersions(dxexonmo0.01, quiet = T)
  dxexonmo0.01 <- testForDEU(dxexonmo0.01, reducedModel =~sample + exon)
})

dxexonredmo0.01 <- DEXSeqResults(dxexonmo0.01, independentFiltering = F)
qval_mor_exon_dex0.01 <- perGeneQValue(dxexonredmo0.01)
dxr.g_mor_exon_dex0.01 <-
data.frame(gene=names(qval_mor_exon_dex0.01),qval_mor_exon_dex0.01)
subset(dxr.g_mor_exon_dex0.01, qval_mor_exon_dex0.01 < 0.05)

### G morhua 1 uM DEU
system.time({
  dxexonmo1 <- estimateSizeFactors(dxexonmo1)
  dxexonmo1 <- estimateDispersions(dxexonmo1, quiet = T)
  dxexonmo1 <- testForDEU(dxexonmo1, reducedModel =~sample + exon)
})

dxexonredmo1 <- DEXSeqResults(dxexonmo1, independentFiltering = F)
qval_mor_exon_dex1 <- perGeneQValue(dxexonredmo1)
dxr.g_mor_exon_dex1 <-
data.frame(gene=names(qval_mor_exon_dex1),qval_mor_exon_dex1)
subset(dxr.g_mor_exon_dex1, qval_mor_exon_dex1 < 0.05)

### G morhua linear DEU
system.time({
  dexex_num_mo <- estimateSizeFactors(dexex_num_mo)
  dexex_num_mo <- estimateDispersions(dexex_num_mo, quiet = T)
  dexex_num_mo <- testForDEU(dexex_num_mo, reducedModel =~sample + exon)
})
dexex_num_mo_res <- DEXSeqResults(dexex_num_mo, independentFiltering = F)
qvalex_mo_dex <- perGeneQValue(dexex_num_mo_res)
dxr.gex_mo_dex <- data.frame(gene=names(qvalex_mo_dex),qvalex_mo_dex)
abcd <- subset(dxr.gex_mo_dex, qvalex_mo_dex < 0.05)
### csv (or txt) files can be written for these in case analyses are not all completed in
### 1 working day. Some of these functions (G morhua exon-based) took several hours
### to run on an M1 MacBook Pro locally

################### Step 4 ##############################
### PCA, then UpSet
# C elegans PCA
vsd <- vst(dds)
pcaData_elegans <- plotPCA(vsd, "Group", returnData = T)
percentVarEle <- round(100 * attr(pcaData_elegans, "percentVar"))
hull.data <- do.call(rbind, lapply(split(pcaData_elegans, list(pcaData_elegans$Group)),
                                   function(x) x[chull(x[,c("PC1", "PC2")]),]))
ggplot(pcaData_elegans, aes(x = PC1, y= PC2, color=Group)) +
  geom_polygon(data = hull.data, alpha = 0) +
   geom_point(size=3) +
  xlab(paste0("PC1: ",percentVarEle[1],"% variance")) +
  ylab(paste0("PC2: ",percentVarEle[2],"% variance")) +
  theme_classic()


# G morhua PCA
vsdmo <- vst(dds2)
pcaData <- plotPCA(vsdmo, "Group", returnData = T)
percentVar <- round(100 * attr(pcaData, "percentVar"))
hull.datamo <- do.call(rbind, lapply(split(pcaData, list(pcaData$Group)),
                                   function(x) x[chull(x[,c("PC1", "PC2")]),]))
ggplot(pcaData, aes(PC1, PC2, color=Group)) +
  geom_point(size=3) +
  xlab(paste0("PC1: ",percentVar[1],"% variance")) +
  ylab(paste0("PC2: ",percentVar[2],"% variance")) +
  theme_classic()

### First, load all sig gene csv files (only necessary if all objects not still loaded)
### only done for C elegans because G morhua has few unique sig features
library(UpSetR)
```

```
sample.data.elegans <- read.table("transcript2gene.txt") ### this is the reference we created in
the linux environment that has all genes contained in the annotation file
names(sample.data.elegans) <- c("WBGene","TranscriptID")
sample.data.elegans$TranscriptID <- str_remove(sample.data.elegans$TranscriptID, "^Name=")
sample.data.elegans$WBGene <- str_remove(sample.data.elegans$WBGene,
"^Parent=Gene:")
res5df1 <- read.csv("res5elegans.csv")
res20df3 <- read.csv("Allsiggenesres20.csv")
resnumalphadf<- read.csv("resnumalphadf.csv")
dxr.gnum_sig <- read.csv("elegansDEXseqNUM.csv")
dxrgex05sig <- read.csv("elegansexon_05.csv")
dxrgex020sig <- read.csv("elegansexon_020.csv")
dxrgexnumsig <- read.csv("elegansexon_NUM.csv")


ups_elegans <- data.frame(gene=unique(sample.data.elegans$WBGene),
                      `Gene_0vs5`=0, `Gene_0vs20`=0, `Gene_Linear`=0,`Transcript_Linear`=0,
                      `Exon_0vs5`=0,`Exon_0vs20`=0,`Exon_Linear`=0)
ups_elegans[ups_elegans$gene %in% res5df1$X,2] <- 1
ups_elegans[ups_elegans$gene %in% res20df3$X,3] <- 1
ups_elegans[ups_elegans$gene %in% resnumalphadf$X,4] <- 1
ups_elegans[ups_elegans$gene %in% dxr.gnum_sig$X,5] <- 1
ups_elegans[ups_elegans$gene %in% dxrgex05sig$X,6] <- 1
ups_elegans[ups_elegans$gene %in% dxrgex020sig$X,7] <- 1
ups_elegans[ups_elegans$gene %in% dxrgexnumsig$X,8] <- 1

upset(ups_elegans, nset=7, order.by = "freq", point.size = 3, text.scale = 1.3,
    sets.bar.color =
c("lightsalmon","lightsalmon","firebrick1","firebrick1","olivedrab1","olivedrab1","lightsalmon"),
    queries = list(list(query = intersects, params = list("Set1","Set2"), color = "red", active = T)))


### Hope you enjoyed. Any comments/improvements/criticisms welcome : D
```