```
# Version info: R 4.2.2, Biobase 2.58.0, GEOquery 2.66.0, limma 3.54.0
################################################################
#   Differential expression analysis with DESeq2
library(DESeq2)

# load counts table from GEO
urld <- "https://www.ncbi.nlm.nih.gov/geo/download/?format=file&type=rnaseq_counts"
path <- paste(urld, "acc=GSE264630",
"file=GSE264630_raw_counts_GRCh38.p13_NCBI.tsv.gz", sep="&");
tbl <- as.matrix(data.table::fread(path, header=T, colClasses="integer"), rownames="GeneID")

# load gene annotations
apath <- paste(urld, "type=rnaseq_counts", "file=Human.GRCh38.p13.annot.tsv.gz", sep="&")
annot <- data.table::fread(apath, header=T, quote="", stringsAsFactors=F, data.table=F)
rownames(annot) <- annot$GeneID

# sample selection
gsms <- "2220011X2X121XX1X21221X21X2"
sml <- strsplit(gsms, split="")[[1]]

# filter out excluded samples (marked as "X")
sel <- which(sml != "X")
sml <- sml[sel]
tbl <- tbl[ ,sel]

# group membership for samples
gs <- factor(sml)
groups <- make.names(c("control","DMSO","LLY283"))
levels(gs) <- groups
sample_info <- data.frame(Group = gs, row.names = colnames(tbl))

# pre-filter low count genes
# keep genes with at least N counts > 10, where N = size of smallest group
keep <- rowSums( tbl >= 10 ) >= min(table(gs))
tbl <- tbl[keep, ]

ds <- DESeqDataSetFromMatrix(countData=tbl, colData=sample_info, design= ~Group)

ds <- DESeq(ds, test="LRT", reduced = ~ 1)  # Use LRT for all-around gene ranking

# extract results for top genes table
r <- results (ds, alpha=0.05, pAdjustMethod ="fdr")

tT <- r[order(r$padj)[1:250],]
```

```
tT <- merge(as.data.frame(tT), annot, by=0, sort=F)

tT <- subset(tT, select=c("GeneID","padj","pvalue","stat","baseMean","Symbol","Description"))
write.table(tT, file=stdout(), row.names=F, sep="\t")

plotDispEsts(ds, main="GSE264630 Dispersion Estimates")

# create histogram plot of p-values
hist(r$padj, breaks=seq(0, 1, length = 21), col = "grey", border = "white",
        xlab = "", ylab = "", main = "GSE264630 Frequencies of padj-values")

# Wald test to obtain contrast-specific results
ds <- DESeq(ds, test="Wald", sfType="poscount")
r <- results (ds, contrast=c("Group", groups[1], groups[2]), alpha=0.05, pAdjustMethod = "fdr")

# volcano plot
old.pal <- palette(c("#00BFFF", "#FF3030")) # low-hi colors
par(mar=c(4,4,2,1), cex.main=1.5)
plot(r$log2FoldChange, -log10(r$padj), main=paste(groups[1], "vs", groups[2]),
    xlab="log2FC", ylab="-log10(Padj)", pch=20, cex=0.5)
with(subset(r, padj<0.05 & abs(log2FoldChange) >= 0),
    points(log2FoldChange, -log10(padj), pch=20, col=(sign(log2FoldChange) + 3)/2, cex=1))
legend("bottomleft", title=paste("Padj<", 0.05, sep=""), legend=c("down", "up"), pch=20,col=1:2)

# MD plot
par(mar=c(4,4,2,1), cex.main=1.5)
plot(log10(r$baseMean), r$log2FoldChange, main=paste(groups[1], "vs", groups[2]),
    xlab="log10(mean of normalized counts)", ylab="log2FoldChange", pch=20, cex=0.5)
with(subset(r, padj<0.05 & abs(log2FoldChange) >= 0),
    points(log10(baseMean), log2FoldChange, pch=20, col=(sign(log2FoldChange) + 3)/2,
cex=1))
legend("bottomleft", title=paste("Padj<", 0.05, sep=""), legend=c("down", "up"), pch=20,col=1:2)
abline(h=0)
palette(old.pal) # restore palette

# Venn diagram
library(gplots)
all_res <- list()

ct.names <- resultsNames(ds)[-1] # contrasts names without Intercept
for (ct in ct.names) {
  r <- results(ds, name=ct, alpha=0.05, pAdjustMethod = "fdr")
  all_res[[length(all_res) + 1]] <- rownames(r)[!is.na(r$padj) & r$padj < 0.05 &
abs(r$log2FoldChange) >= 0]
```

```
}
names(all_res) <- ct.names

venn(all_res)

######################################################################
#   General expression data visualization
dat <- log10(counts(ds, normalized = T) + 1) # extract normalized counts

# box-and-whisker plot
lbl <- "log10(raw counts + 1)"
ord <- order(gs)  # order samples by group
palette(c("#1B9E77", "#7570B3", "#E7298A", "#E6AB02", "#D95F02",
          "#66A61E", "#A6761D", "#B32424", "#B324B3", "#666666"))
par(mar=c(7,4,2,1))
boxplot(dat[,ord], boxwex=0.6, notch=T, main="GSE264630", ylab="lg(norm.counts)", outline=F,
las=2, col=gs[ord])
legend("topleft", groups, fill=palette(), bty="n")

# UMAP plot (multi-dimensional scaling)
library(umap)
dat <- dat[!duplicated(dat), ] # first remove duplicates
par(mar=c(3,3,2,6), xpd=TRUE, cex.main=1.5)
ump <- umap(t(dat), n_neighbors = 9, random_state = 123)
plot(ump$layout, main="UMAP plot, nbrs=9", xlab="", ylab="", col=gs, pch=20, cex=1.5)
legend("topright", inset=c(-0.15,0), legend=groups, pch=20,
       col=1:length(groups), title="Group", pt.cex=1.5)
```