Group Project: Tables

The table group

3/26/2019

Contents


```
if (!require("venn")) install.packages("venn", repos='http://cran.us.r-project.org'); library(venn)
```

```
## Loading required package: venn
```

```
if (!require("VennDiagram")) install.packages("VennDiagram", repos='http://cran.us.r-project.org'); libi
```

```
## Loading required package: VennDiagram
```

```
## Warning in library(package, lib.loc = lib.loc, character.only = TRUE,
## logical.return = TRUE, : there is no package called 'VennDiagram'
```

```
##
## The downloaded binary packages are in
##  /var/folders/dr/rlkxn8vx5_700djh1yj20z5m0007_6/T//RtmpFz0773/downloaded_packages
```

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

```
## Loading required package: futile.logger
```

```
if (!require("grid")) install.packages("grid", repos='http://cran.us.r-project.org'); library(grid)
if (!require("gridExtra")) install.packages("gridExtra", repos='http://cran.us.r-project.org'); library
```

```
## Loading required package: gridExtra
```

```
if (!require("dplyr")) install.packages("dplyr, repos='http://cran.us.r-project.org'"); library(dplyr)
```

```
## Loading required package: dplyr
```

```
##
## Attaching package: 'dplyr'
```

```
## The following object is masked from 'package:gridExtra':
##
##     combine
```

```
## The following objects are masked from 'package:stats':
##
##     filter, lag
## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

Loading in Tables

```
bowtie_deseq_table <- read.csv("tables/bowtie2_deseq2_transcript.csv")
bowtie_edgeR_table <- read.csv("tables/Bowtie2EdgeR_McGauley.csv")
bowtie_limma_table <- read.csv("tables/Moore_bowtie_limma.csv")

kallisto_edgeR_table <- read.csv("tables/julie_kallisto_edgeR_final.csv")
kallisto_limma_table <- read.csv("tables/SimpsonTable.csv")
kallisto_deseq_table <- read.csv("tables/kallistoDESeq_Sussman.csv")
kallisto_deseq_table <- kallisto_deseq_table[1:4]


salmon_limma_table <- read.csv("tables/McKinleySalmonLimmaGeneInfo.csv")
salmon_edgeR_table <- read.csv("tables/KE_Salmon_EdgeR_final.csv")
salmon_deseq_table <- read.csv("tables/salmondeseq2.csv")

sailfish_limma_table <- read.csv("tables/Ritter_sailfish_limma.table.csv")
sailfish_edgeR_table <- read.csv("tables/JustinKoss.sailfish.edgeR.updown.csv")
sailfish_deseq_table <- read.csv("tables/Vogel_Sailfish_DESeq2.csv")
```

```
colnames(bowtie_deseq_table)
```

```
## [1] "Align"    "DE"       "Dir"      "GeneName"
```

```
colnames(bowtie_edgeR_table)
```

```
## [1] "Align"    "DE"       "Dir"      "GeneName"
```

```
colnames(bowtie_limma_table)
```

```
## [1] "Align"    "DE"       "Dir"      "GeneName"
```

```
colnames(kallisto_deseq_table)
```

```
## [1] "Align"    "DE"       "Dir"      "GeneName"
```

```
colnames(kallisto_edgeR_table)
```

```
## [1] "Align"    "DE"       "Dir"      "GeneName"
```

```
colnames(kallisto_limma_table)
```

```
## [1] "Align"    "DE"       "Dir"      "GeneName"
```

```
colnames(salmon_deseq_table)
```

```
## [1] "Align"    "DE"       "Dir"      "GeneName"
```

```
colnames(salmon_edgeR_table)
```

```
## [1] "Align"    "DE"       "Dir"      "GeneName"
```

```
colnames(salmon_limma_table)
```

```
## [1] "Align"    "DE"       "Dir"      "GeneName"
```

```
colnames(sailfish_deseq_table)
```

```
## [1] "Align"    "DE"       "Dir"      "GeneName"
```

```
colnames(sailfish_edgeR_table)
```

```
## [1] "Align"    "DE"       "Dir"      "GeneName"
```

```
colnames(sailfish_limma_table)
```

```
## [1] "Align"    "DE"       "Dir"      "GeneName"
```

```
colnames(sailfish_limma_table)[1] <- "Align"
```

Creating a super table

```
complete_table <- rbind(bowtie_deseq_table, bowtie_edgeR_table, bowtie_limma_table, kallisto_edgeR_table
write.csv(complete_table, file ="tables/All_pipelines_complete_table.csv", quote = FALSE)

#confirming number of rows is correct in new table
nrow(bowtie_deseq_table) +nrow(bowtie_edgeR_table) +nrow(bowtie_limma_table) +nrow(kallisto_deseq_table
```

```
## [1] TRUE
```

```
#not getting the right number of responses here, perhaps a capitalization issue
n_distinct(complete_table$Align)
```

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

```
unique(complete_table$Align)
```

```
## [1] bowtie2  kallisto salmon   sailfish
## Levels: bowtie2 kallisto salmon sailfish
```

```
n_distinct(complete_table$DE)
```

```
## [1] 3
```

```
unique(complete_table$DE)
```

```
## [1] DESeq2 edgeR  limma
## Levels: DESeq2 edgeR limma
```

DE work

Preparing individual DE tables

```
DESeq_venn <- filter(complete_table, DE == "DESeq2")

nrow(filter(DESeq_venn, Align == "bowtie2")) #478
```

```
## [1] 478
```

```
nrow(filter(DESeq_venn, Align == "kallisto")) #431
```

```
## [1] 431
```

```
nrow(filter(DESeq_venn, Align == "salmon")) #2
```

```
## [1] 2
```

```
nrow(filter(DESeq_venn, Align == "sailfish")) #645
```

```
## [1] 645
```

```
#again, double checking that these add up to the total number of roles in the complete table
nrow(subset(DESeq_venn, Align == "bowtie2")) +
nrow(subset(DESeq_venn, Align == "kallisto")) +
nrow(subset(DESeq_venn, Align == "salmon")) +
nrow(subset(DESeq_venn, Align == "sailfish")) == nrow(DESeq_venn)
```

```
## [1] TRUE
```

```
edgeR_venn <- filter(complete_table, DE == "edgeR")

nrow(filter(edgeR_venn, Align == "bowtie2")) #537
```

```
## [1] 537
```

```
nrow(filter(edgeR_venn, Align == "kallisto")) #319
```

```
## [1] 319
```

```
nrow(filter(edgeR_venn, Align == "salmon")) #541
```

```
## [1] 541
```

```
nrow(filter(edgeR_venn, Align == "sailfish")) #395
```

```
## [1] 395
```

```
#again, double checking that these add up to the total number of roles in the complete table
nrow(subset(edgeR_venn, Align == "bowtie2")) +
nrow(subset(edgeR_venn, Align == "kallisto")) +
nrow(subset(edgeR_venn, Align == "salmon")) +
nrow(subset(edgeR_venn, Align == "sailfish")) == nrow(edgeR_venn)
```

```
## [1] TRUE
```

```
limma_venn <- filter(complete_table, DE == "limma")

nrow(filter(limma_venn, Align == "bowtie2")) #182
```

```
## [1] 182
```

```
nrow(filter(limma_venn, Align == "kallisto")) #1
```

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

```
nrow(filter(limma_venn, Align == "salmon")) #338
```

```
## [1] 338
```

```
nrow(filter(limma_venn, Align == "sailfish")) #213
```

```
## [1] 213
```

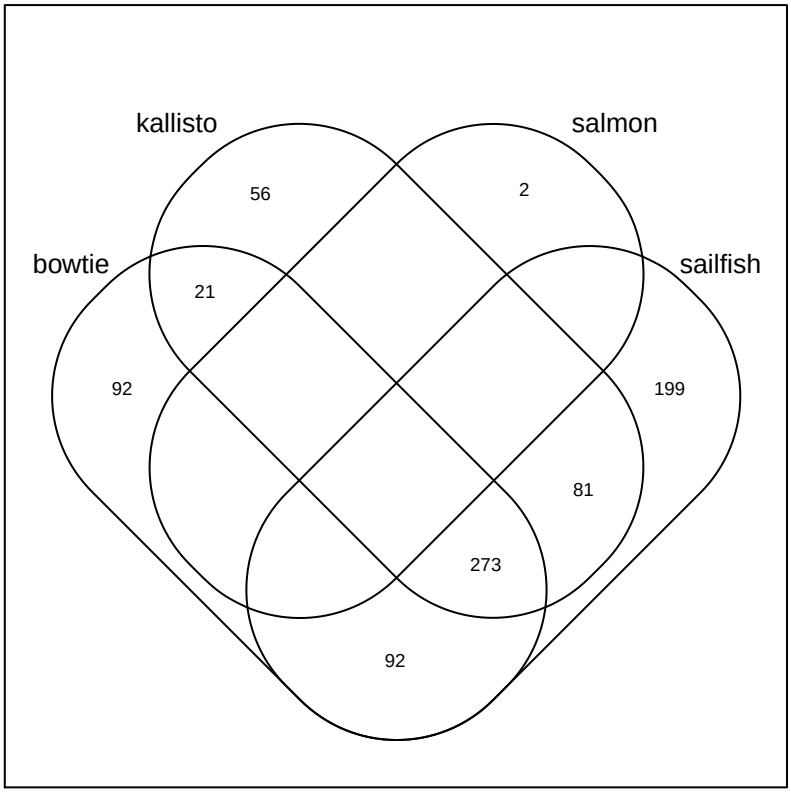
```
#again, double checking that these add up to the total number of roles in the complete table
nrow(subset(limma_venn, Align == "bowtie2")) +
nrow(subset(limma_venn, Align == "kallisto")) +
nrow(subset(limma_venn, Align == "salmon")) +
nrow(subset(limma_venn, Align == "sailfish")) == nrow(limma_venn)
```

```
## [1] TRUE
```

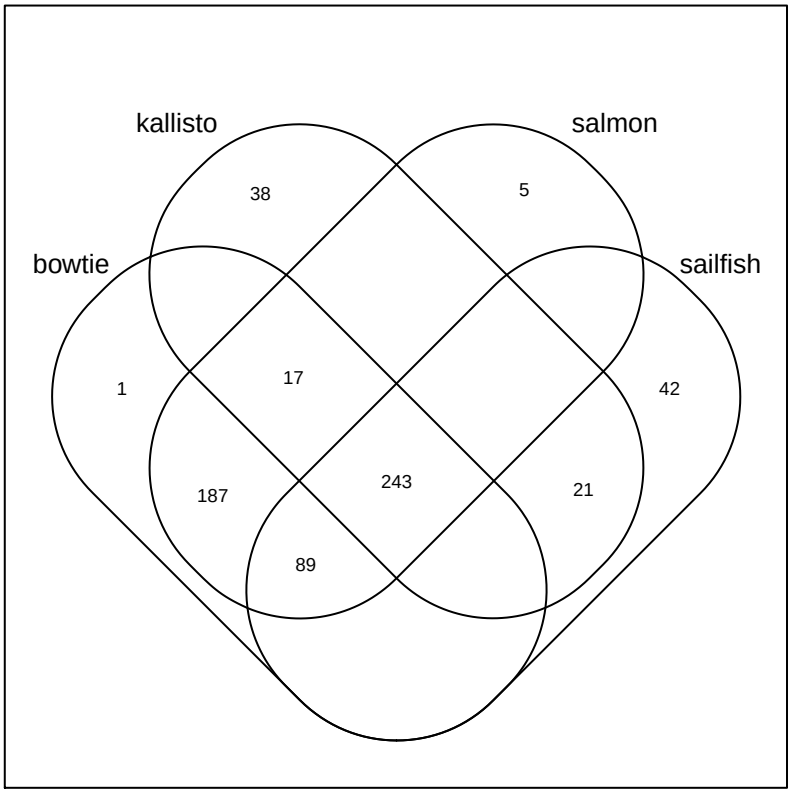
Venn Diagram Work: DE

Trying the package venn: success!

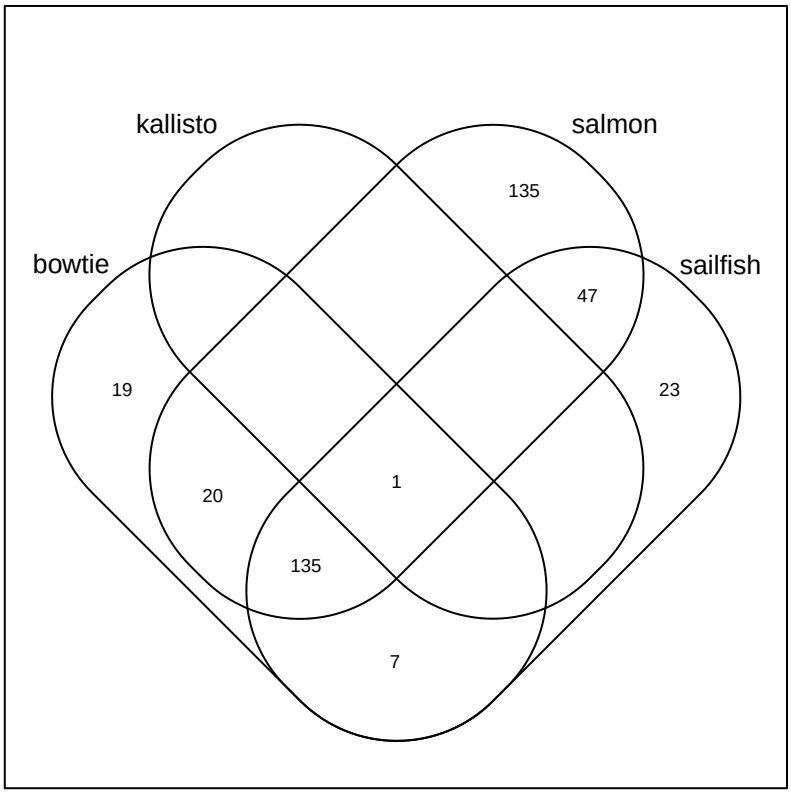
```
library(venn)
#DESeq
venn_DESeq_list <- list(bowtie = subset(DESeq_venn, Align == "bowtie2")$GeneName, kallisto = subset(DES
venn_DESeq_plot <- venn(venn_DESeq_list, show.plot=TRUE)
```




```
#edgeR
venn_edgeR_list <- list(bowtie = subset(edgeR_venn, Align == "bowtie2")$GeneName, kallisto = subset(edg
venn_edgeR_plot <- venn(venn_edgeR_list, show.plot=TRUE)
```




```
#limma
venn_limma_list <- list(bowtie = subset(limma_venn, Align == "bowtie2")$GeneName, kallisto = subset(limm
venn_limma_plot <- venn(venn_limma_list, show.plot=TRUE)
```



Aligner work

Preparing individual aligner tables

```
bowtie2_venn <- filter(complete_table, Align == "bowtie2")

nrow(filter(bowtie2_venn, DE == "DESeq2")) #478
```

```
## [1] 478
```

```
nrow(filter(bowtie2_venn, DE == "edgeR")) #537
```

```
## [1] 537
```

```
nrow(filter(bowtie2_venn, DE == "limma")) #182
```

```
## [1] 182
```

```
#again, double checking that these add up to the total number of roles in the complete table
nrow(subset(bowtie2_venn, DE == "DESeq2")) +
nrow(subset(bowtie2_venn, DE == "edgeR")) +
nrow(subset(bowtie2_venn, DE == "limma")) == nrow(bowtie2_venn)
```

```
## [1] TRUE
```

```
kallisto_venn <- filter(complete_table, Align == "kallisto")

nrow(filter(kallisto_venn, DE == "DESeq2")) #431
```

```
## [1] 431
```

```
nrow(filter(kallisto_venn, DE == "edgeR")) #319
```

```
## [1] 319
```

```
nrow(filter(kallisto_venn, DE == "limma")) #1
```

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

```
#again, double checking that these add up to the total number of roles in the complete table
nrow(subset(kallisto_venn, DE == "DESeq2")) +
nrow(subset(kallisto_venn, DE == "edgeR")) +
nrow(subset(kallisto_venn, DE == "limma")) == nrow(kallisto_venn)
```

```
## [1] TRUE
```

```
salmon_venn <- filter(complete_table, Align == "salmon")

nrow(filter(salmon_venn, DE == "DESeq2")) #2
```

```
## [1] 2
```

```
nrow(filter(salmon_venn, DE == "edgeR")) #541
```

```
## [1] 541
```

```
nrow(filter(salmon_venn, DE == "limma")) #338
```

```
## [1] 338
```

```
#again, double checking that these add up to the total number of roles in the complete table
nrow(subset(salmon_venn, DE == "DESeq2")) +
nrow(subset(salmon_venn, DE == "edgeR")) +
nrow(subset(salmon_venn, DE == "limma")) == nrow(salmon_venn)
```

```
## [1] TRUE
```

```
sailfish_venn <- filter(complete_table, Align == "sailfish")

nrow(filter(sailfish_venn, DE == "DESeq2")) #645
```

```
## [1] 645
```

```
nrow(filter(sailfish_venn, DE == "edgeR")) #395
```

```
## [1] 395
```

```
nrow(filter(sailfish_venn, DE == "limma")) #213
```

```
## [1] 213
```

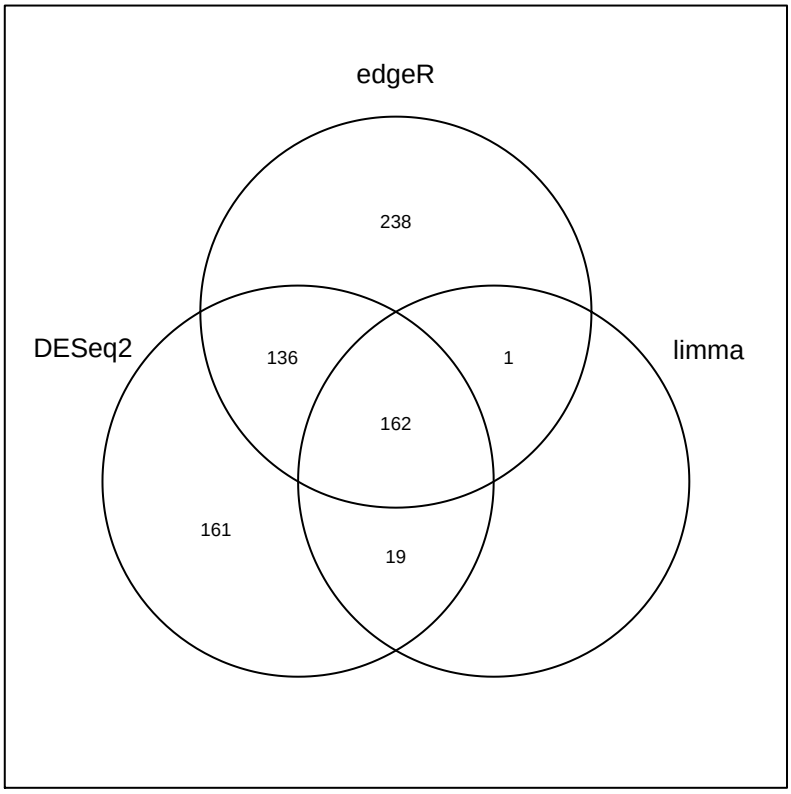
```
#again, double checking that these add up to the total number of roles in the complete table
nrow(subset(sailfish_venn, DE == "DESeq2")) +
nrow(subset(sailfish_venn, DE == "edgeR")) +
nrow(subset(sailfish_venn, DE == "limma")) == nrow(sailfish_venn)
```

```
## [1] TRUE
```

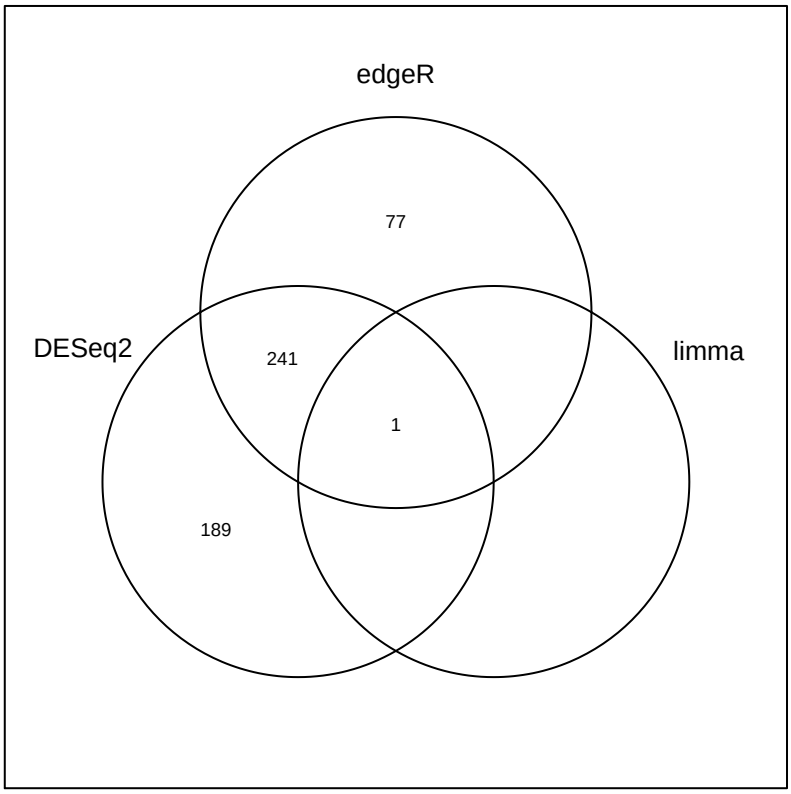
Venn Diagram Work: Aligners

Trying the package venn: success!

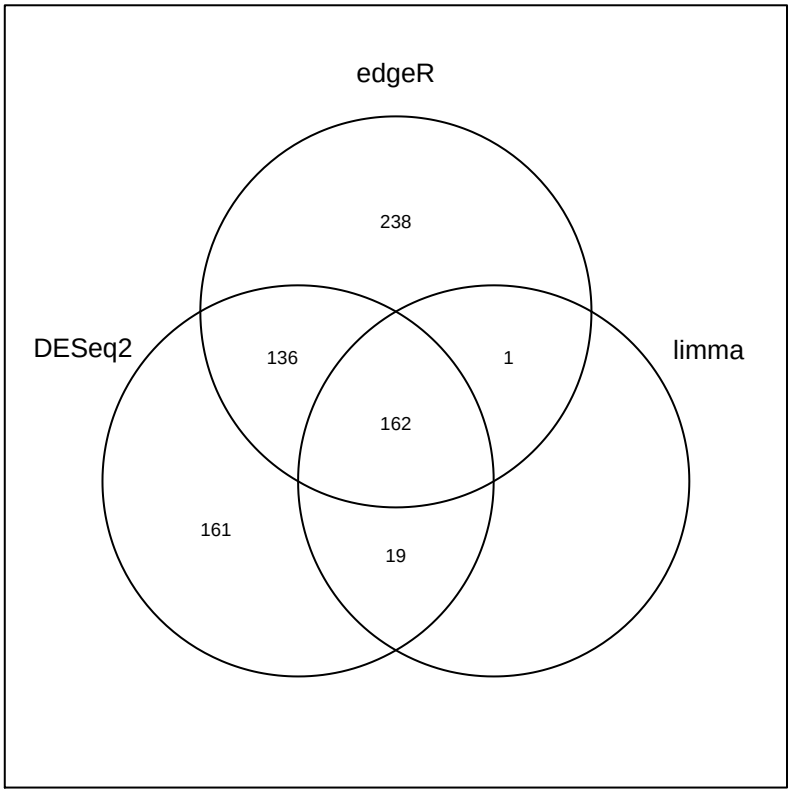
```
#bowtie2
venn_bowtie2_list <- list(DESeq2 = subset(bowtie2_venn, DE == "DESeq2")$GeneName, edgeR = subset(bowtie
venn_bowtie2_plot <- venn(venn_bowtie2_list, show.plot=TRUE)
```



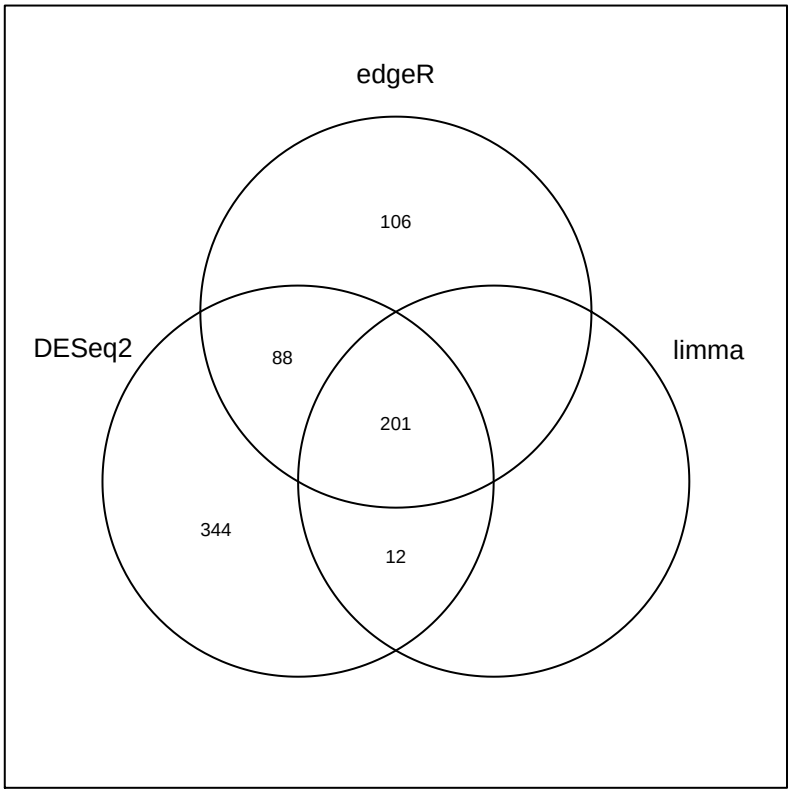

```
#kallisto
venn_kallisto_list <- list(DESeq2 = subset(kallisto_venn, DE == "DESeq2")$GeneName, edgeR = subset(kalli
venn_kallisto_plot <- venn(venn_kallisto_list, show.plot=TRUE)
```



```
#salmon
venn_salmon_list <- list(DESeq2 = subset(salmon_venn, DE == "DESeq2")$GeneName, edgeR = subset(salmon_v
venn_bowtie2_plot <- venn(venn_bowtie2_list, show.plot=TRUE)
```




```
#sailfish
venn_sailfish_list <- list(DESeq2 = subset(sailfish_venn, DE == "DESeq2")$GeneName, edgeR = subset(saili
venn_sailfish_plot <- venn(venn_sailfish_list, show.plot=TRUE)
```



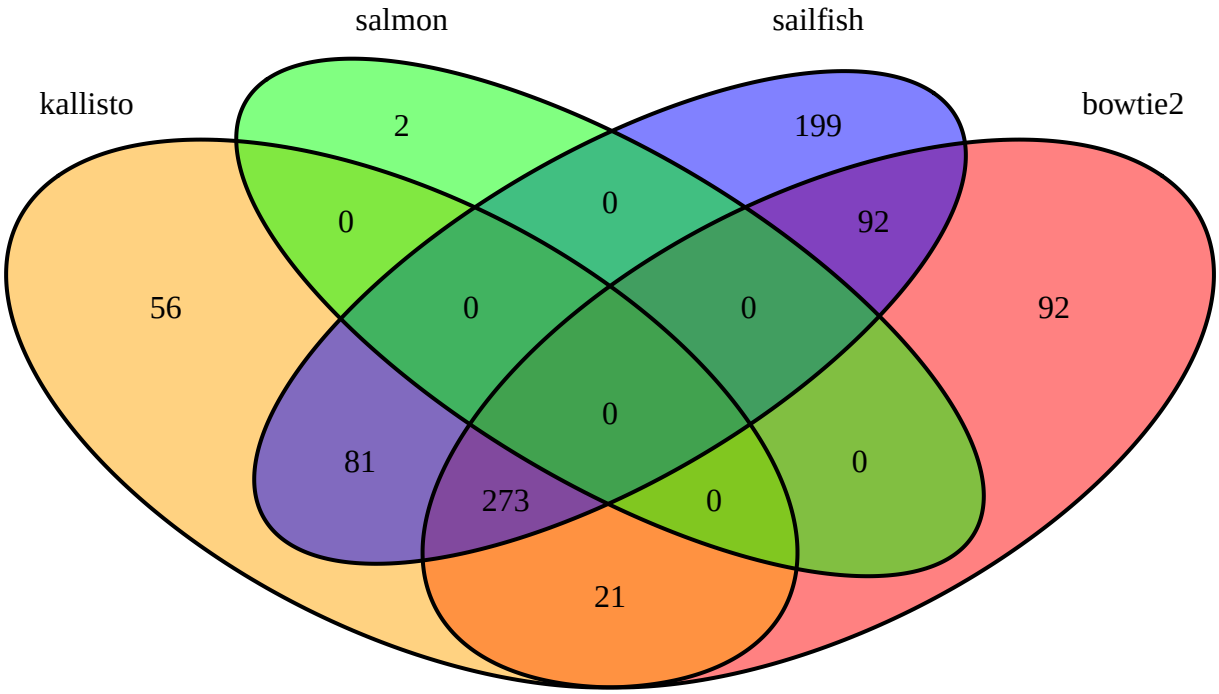
Julie's attempt at venn diagrams

I had to run the code above to get the correct numbers for each overlap. So that's still necessary. But this is another way to display the info but with color :) !!!

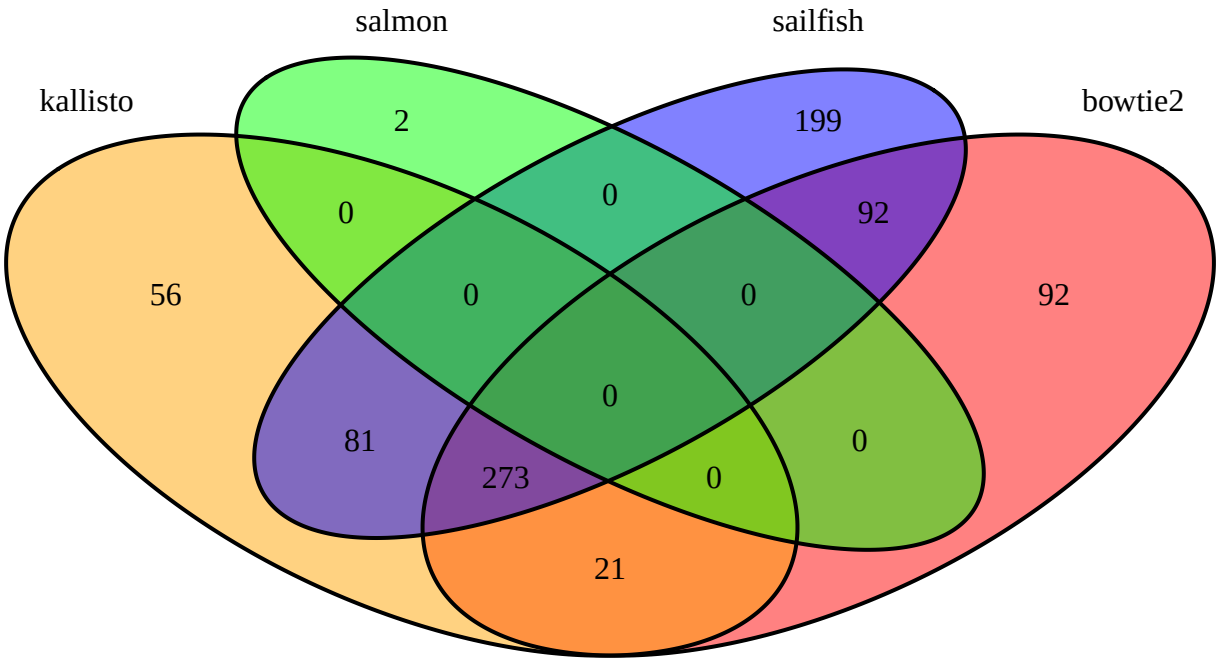
Venn diagrams for DE

```
DESeq2.venn.plot <- draw.quad.venn(
  area1 = 431,
  area2 = 478,
  area3 = 2,
  area4 = 645,
  n12 = 294,
  n13 = 0,
  n14 = 354,
  n23 = 0,
  n24 = 365,
  n34 = 0,
  n123 = 0,
  n124 = 273,
  n134 = 0,
  n234 = 0,
  n1234 = 0,
  category = c("kallisto", "bowtie2", "salmon", "sailfish"),
  fill = c("orange", "red", "green", "blue"),
  cex = 1,
  cat.cex = 1
```

```
)
```




```
require(gridExtra)
grid.arrange(gTree(children=DESeq2.venn.plot), top="Differential Expression using DESeq2")
```



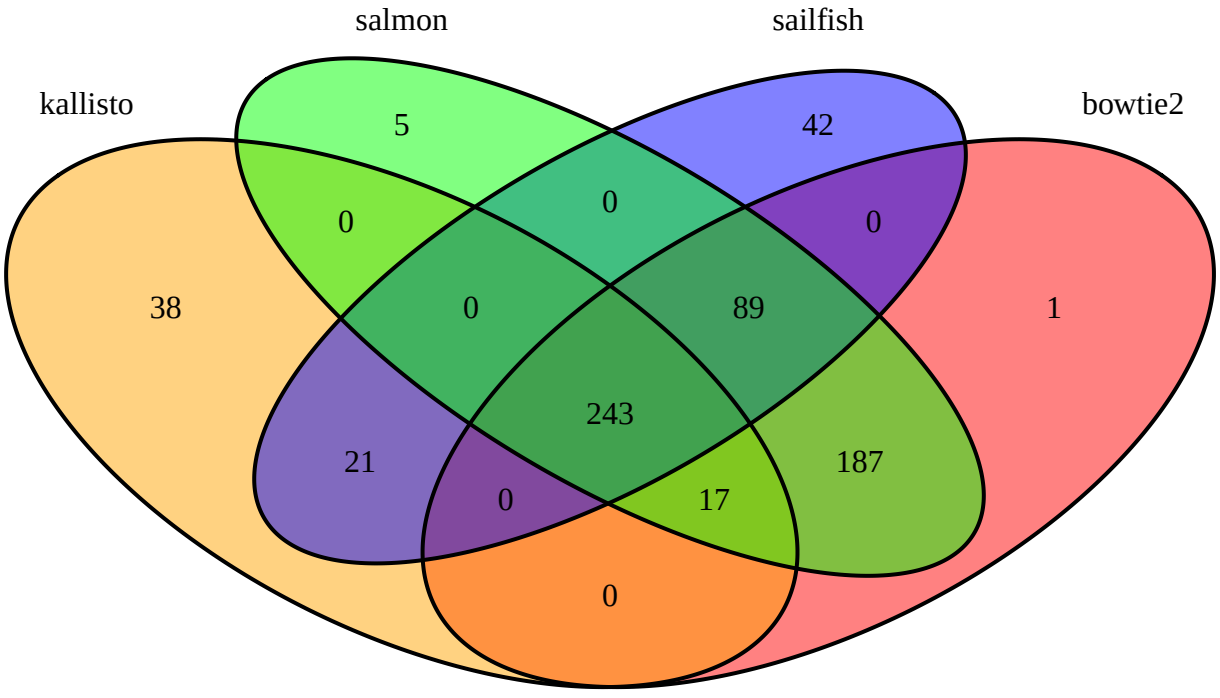

```
#saves plot as png
dev.copy(png,'venn_figs/deseq2venn.png')
```

```
## quartz_off_screen
##                 3
```

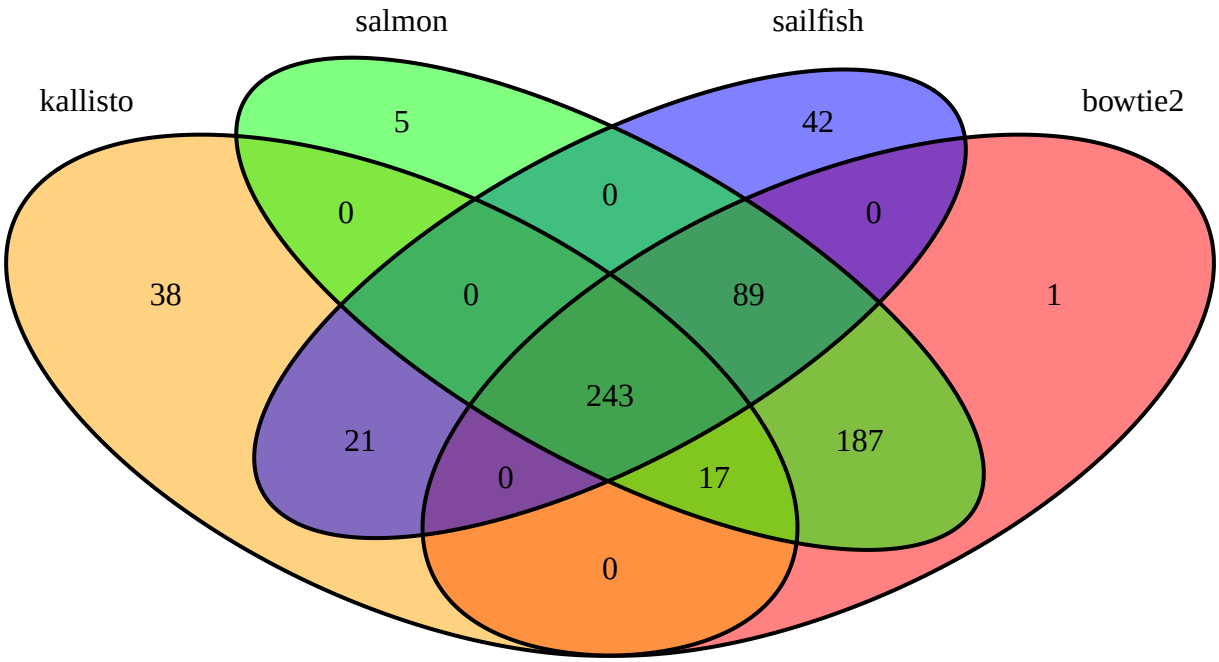
```
dev.off()
```

```
## pdf
##   2
```

```
edgeR.venn.plot <- draw.quad.venn(
  area1 = 319,
  area2 = 537,
  area3 = 541,
  area4 = 395,
  n12 = 260,
  n13 = 260,
  n14 = 264,
  n23 = 536,
  n24 = 332,
  n34 = 332,
  n123 = 260,
  n124 = 243,
  n134 = 243,
  n234 = 332,
  n1234 = 243,
  category = c("kallisto", "bowtie2", "salmon", "sailfish"),
  fill = c("orange", "red", "green", "blue"),
  cex = 1,
  cat.cex = 1
)
```




```
require(gridExtra)
grid.arrange(gTree(children=edgeR.venn.plot), top="Differential Expression using edgeR")
```




```
#saves plot as png
dev.copy(png,'venn_figs/edgrvenn.png')
```

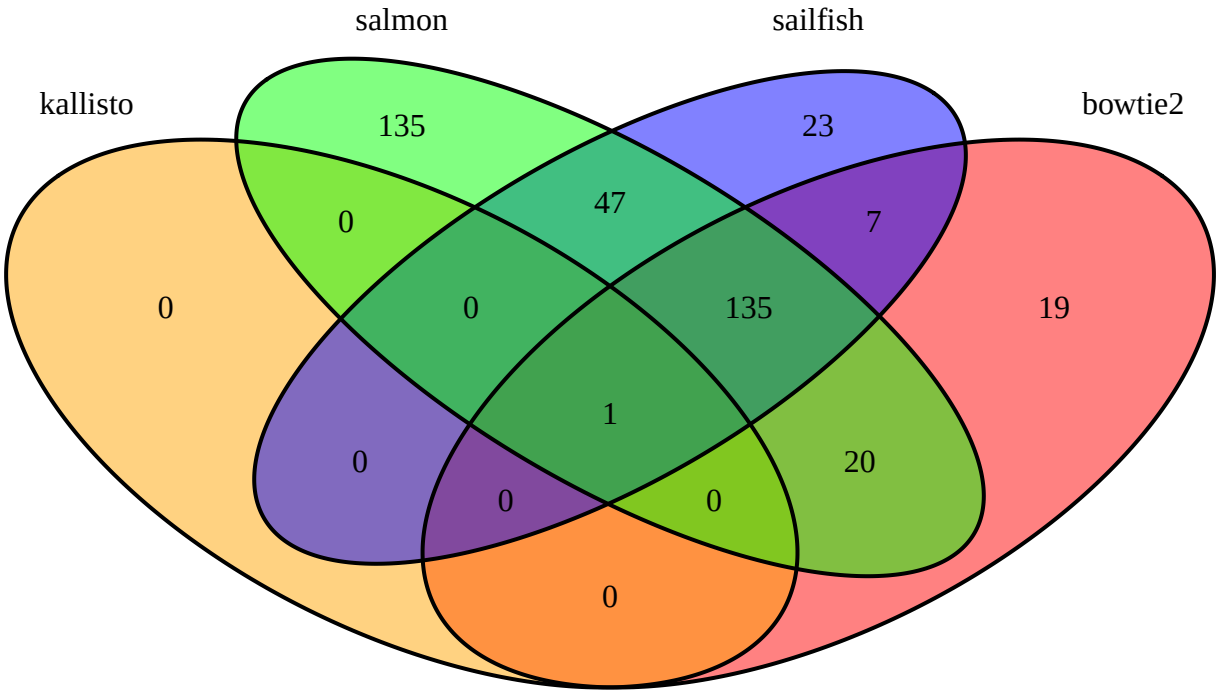
```
## quartz_off_screen
##                 3
```

```
dev.off()
```

```
## pdf
##   2
```

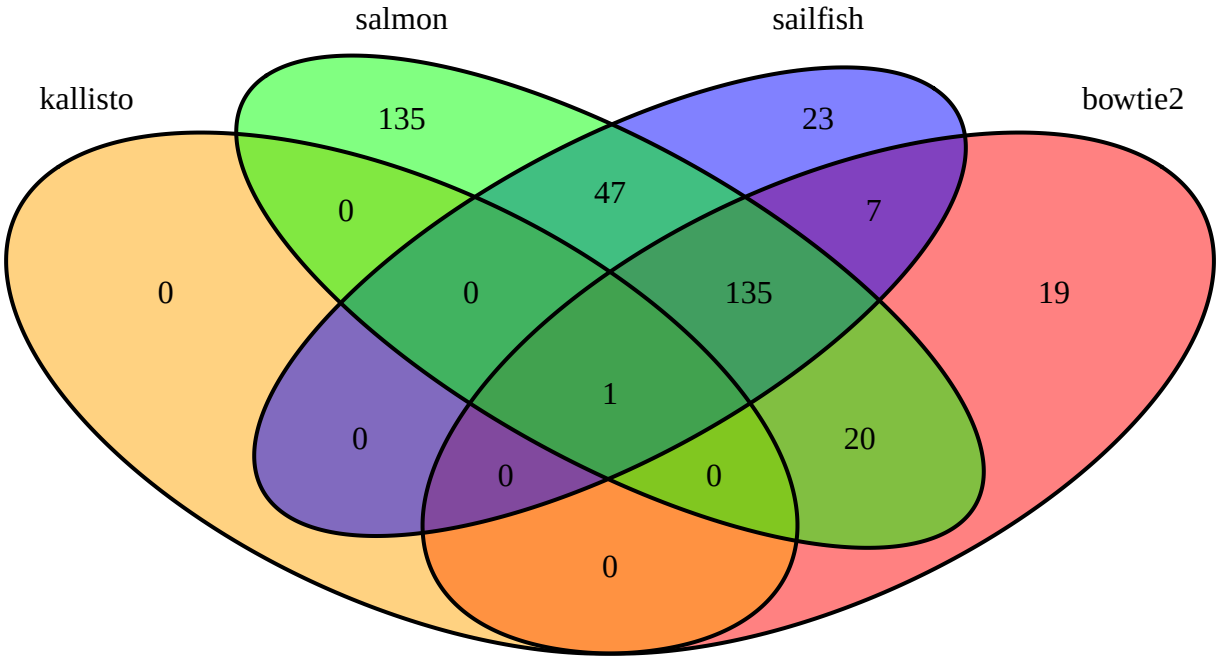
```
limma.venn.plot <- draw.quad.venn(
  area1 = 1,
  area2 = 182,
  area3 = 338,
  area4 = 213,
  n12 = 1,
  n13 = 1,
  n14 = 1,
  n23 = 156,
  n24 = 143,
  n34 = 183,
  n123 = 1,
  n124 = 1,
  n134 = 1,
  n234 = 136,
  n1234 = 1,
  category = c("kallisto", "bowtie2", "salmon", "sailfish"),
  fill = c("orange", "red", "green", "blue"),
  cex = 1,
  cat.cex = 1
```

```
)
```




```
require(gridExtra)
grid.arrange(gTree(children=limma.venn.plot), top="Differential Expression using limma")
```

Differential Expression using limma




```
#saves plot as png
dev.copy(png,'venn_figs/limmavenn.png')
```

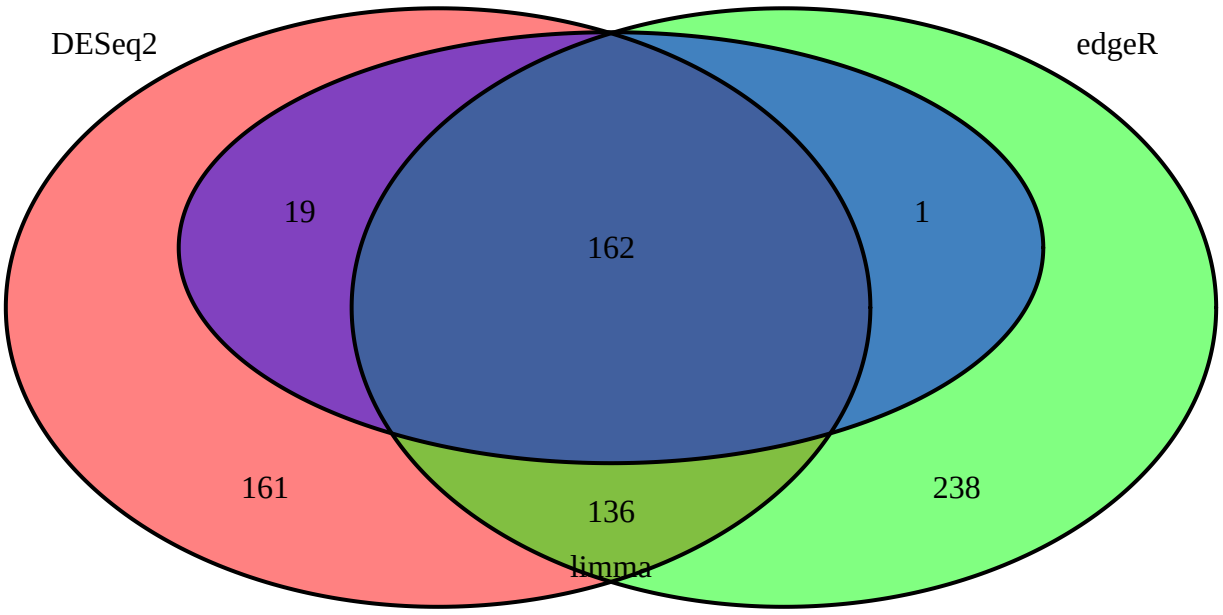
```
## quartz_off_screen
##                 3
```

```
dev.off()
```

```
## pdf
##   2
```

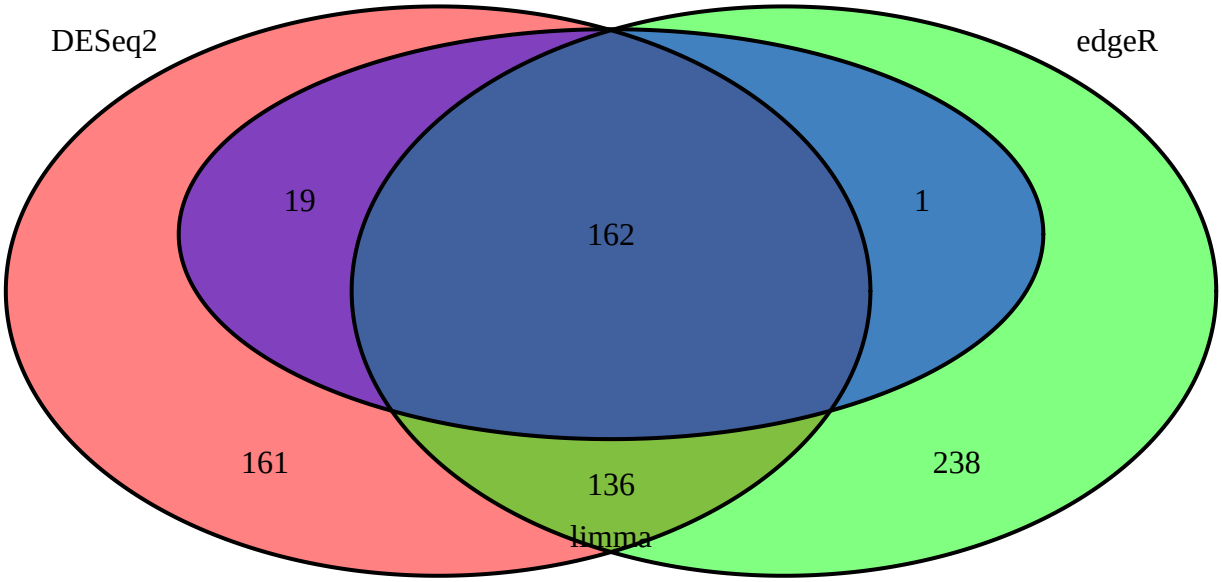
Venn diagrams for aligners

```
bowtie2.venn.plot = draw.triple.venn(
  area1 = 478,
  area2 = 537,
  area3 = 182,
  n12 = 298,
  n23 = 163,
  n13 = 181,
  n123 = 162,
  category = c("DESeq2", "edgeR", "limma"),
  fill = c("red", "green", "blue"),
  cex = 1,
  cat.cex = 1
)
```




```
require(gridExtra)
grid.arrange(gTree(children=bowtie2.venn.plot), top="Differential Expression using bowtie2")
```

Differential Expression using bowtie2



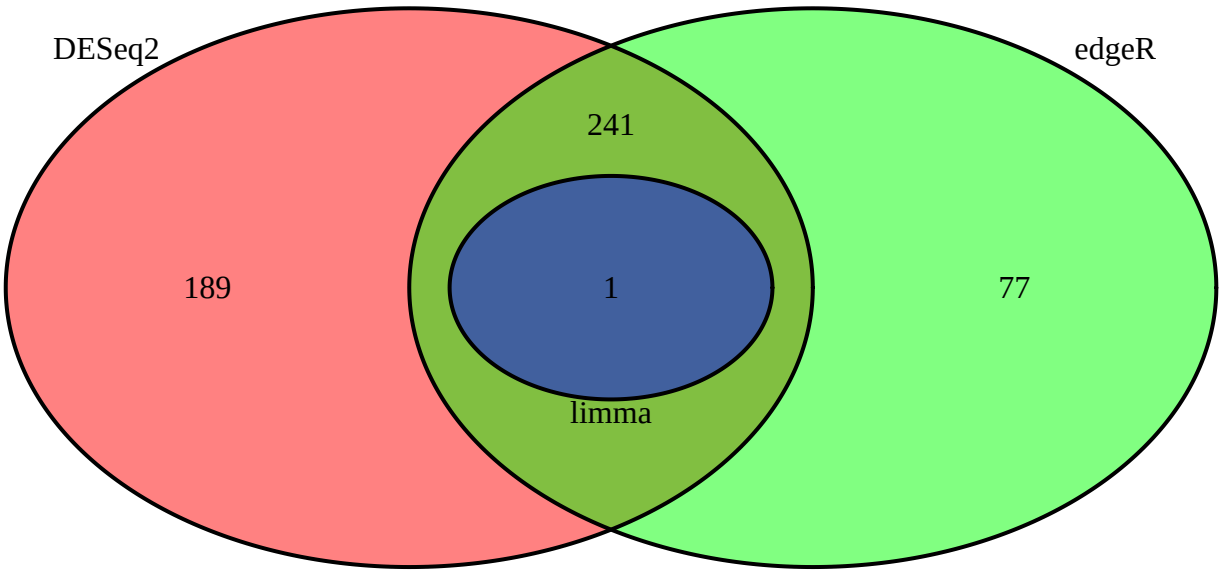

```
#saves plot as png
dev.copy(png,'venn_figs/bowtievenn.png')
```

```
## quartz_off_screen
##                 3
```

```
dev.off()
```

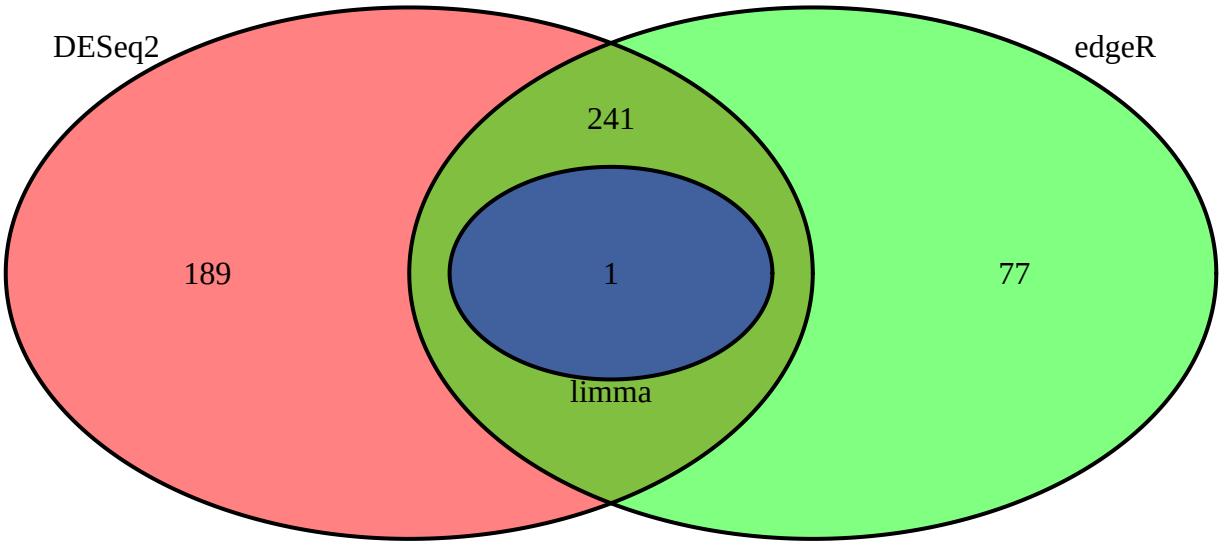
```
## pdf
##   2
```

```
kallisto.venn.plot = draw.triple.venn(
  area1 = 431,
  area2 = 319,
  area3 = 1,
  n12 = 242,
  n23 = 1,
  n13 = 1,
  n123 = 1,
  category = c("DESeq2", "edgeR", "limma"),
  fill = c("red", "green", "blue"),
  cex = 1,
  cat.cex = 1
)
```




```
require(gridExtra)
grid.arrange(gTree(children=kallisto.venn.plot), top="Differential Expression using kallisto")
```

Differential Expression using kallisto



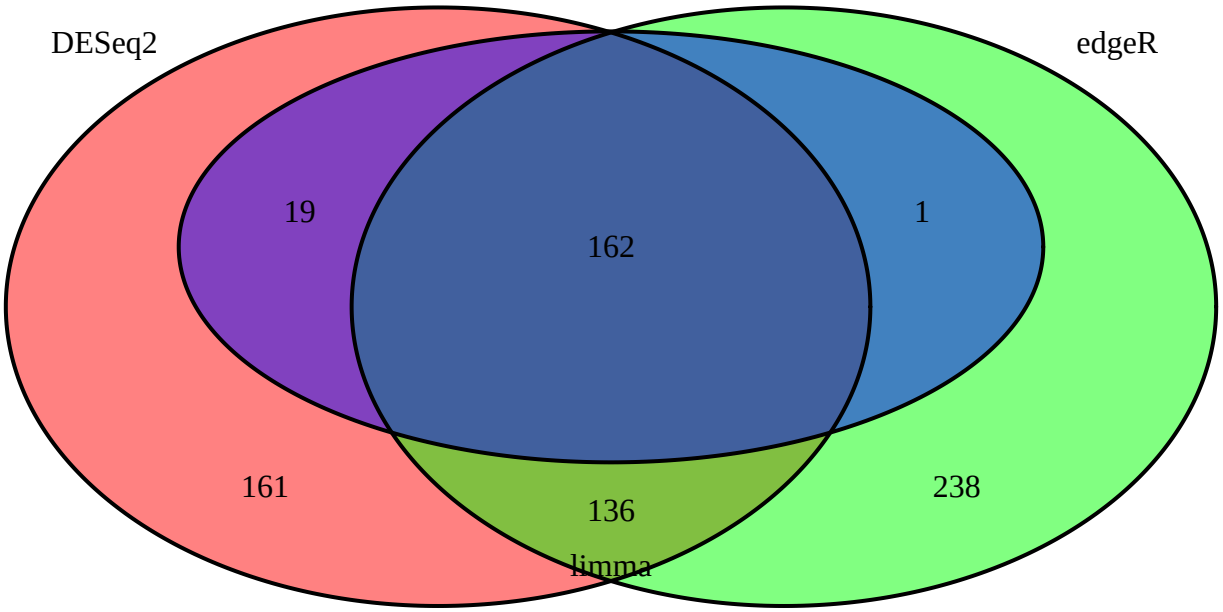

```
#saves plot as png
dev.copy(png,'venn_figs/kallistovenn.png')
```

```
## quartz_off_screen
##                 3
```

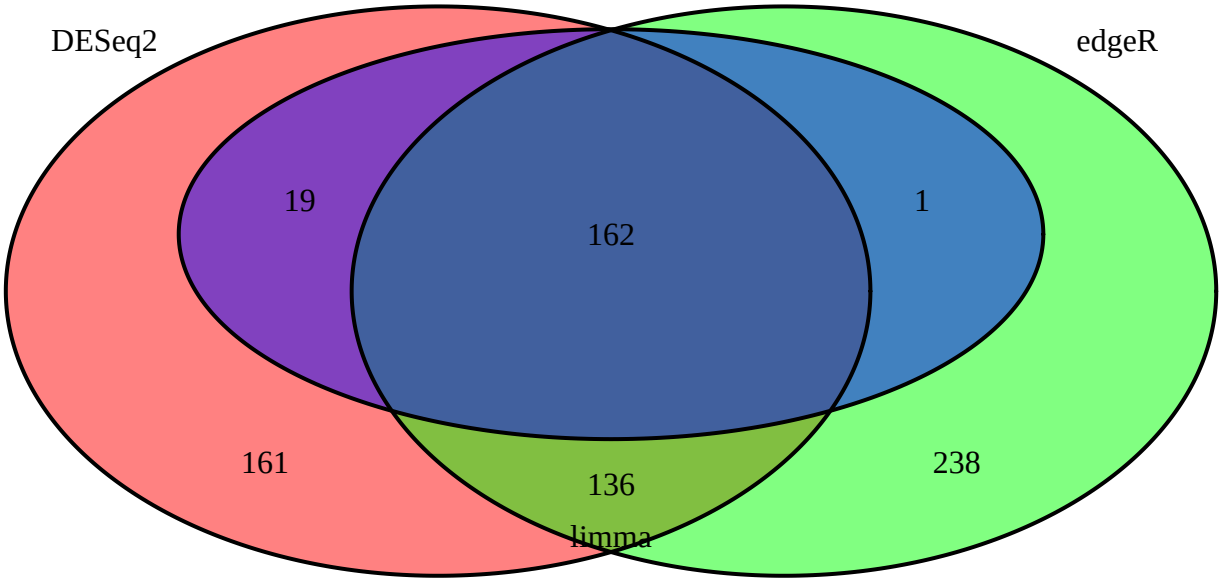
```
dev.off()
```

```
## pdf
##   2
```

```
salmon.venn.plot = draw.triple.venn(
  area1 = 478,
  area2 = 537,
  area3 = 182,
  n12 = 298,
  n23 = 163,
  n13 = 181,
  n123 = 162,
  category = c("DESeq2", "edgeR", "limma"),
  fill = c("red", "green", "blue"),
  cex = 1,
  cat.cex = 1
)
```




```
require(gridExtra)
grid.arrange(gTree(children=salmon.venn.plot), top="Differential Expression using salmon")
```



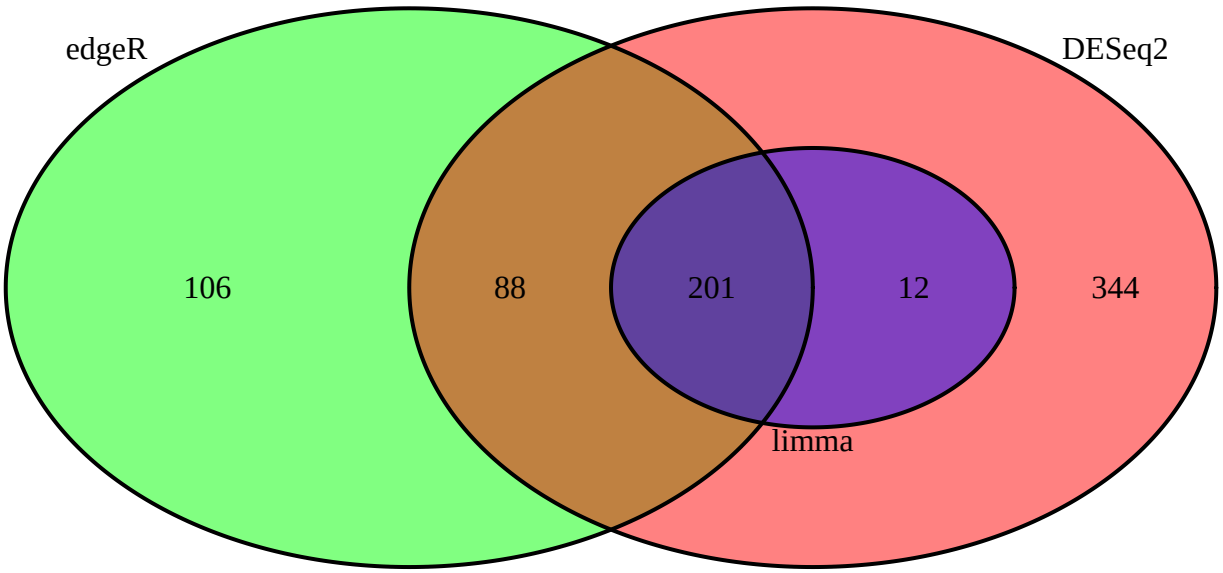

```
#saves plot as png
dev.copy(png,'venn_figs/salmonvenn.png')
```

```
## quartz_off_screen
##                 3
```

```
dev.off()
```

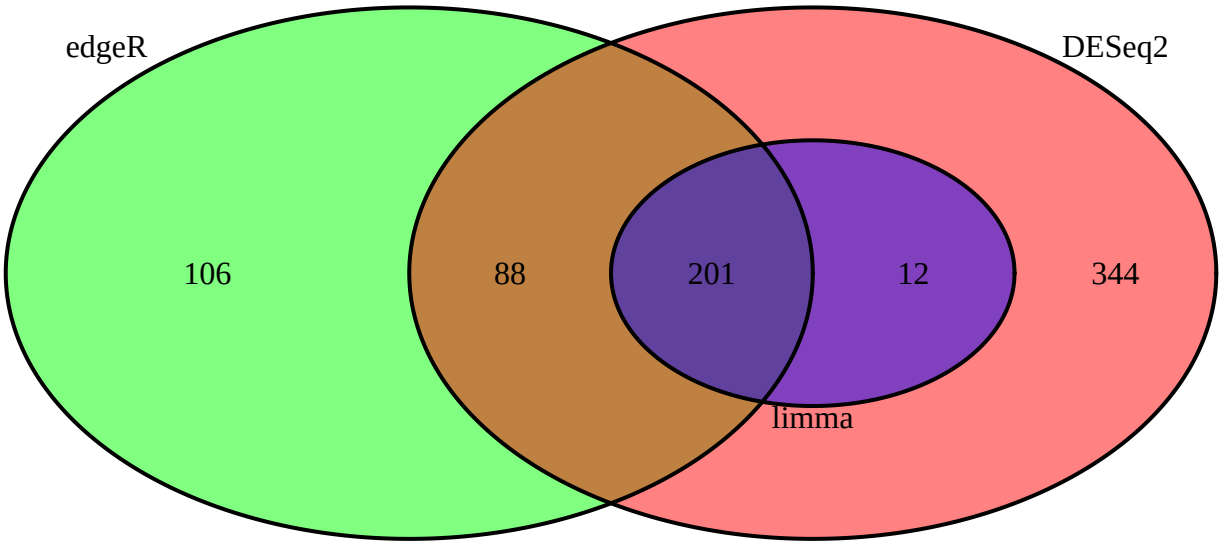
```
## pdf
##   2
```

```
sailfish.venn.plot = draw.triple.venn(
  area1 = 645,
  area2 = 395,
  area3 = 213,
  n12 = 289,
  n23 = 201,
  n13 = 213,
  n123 = 201,
  category = c("DESeq2", "edgeR", "limma"),
  fill = c("red", "green", "blue"),
  cex = 1,
  cat.cex = 1
) -> sailfishvenn
```




```
require(gridExtra)
grid.arrange(gTree(children=sailfish.venn.plot), top="Differential Expression using sailfish")
```

Differential Expression using sailfish




```
#saves plot as png
dev.copy(png,'venn_figs/sailfishvenn.png')
```

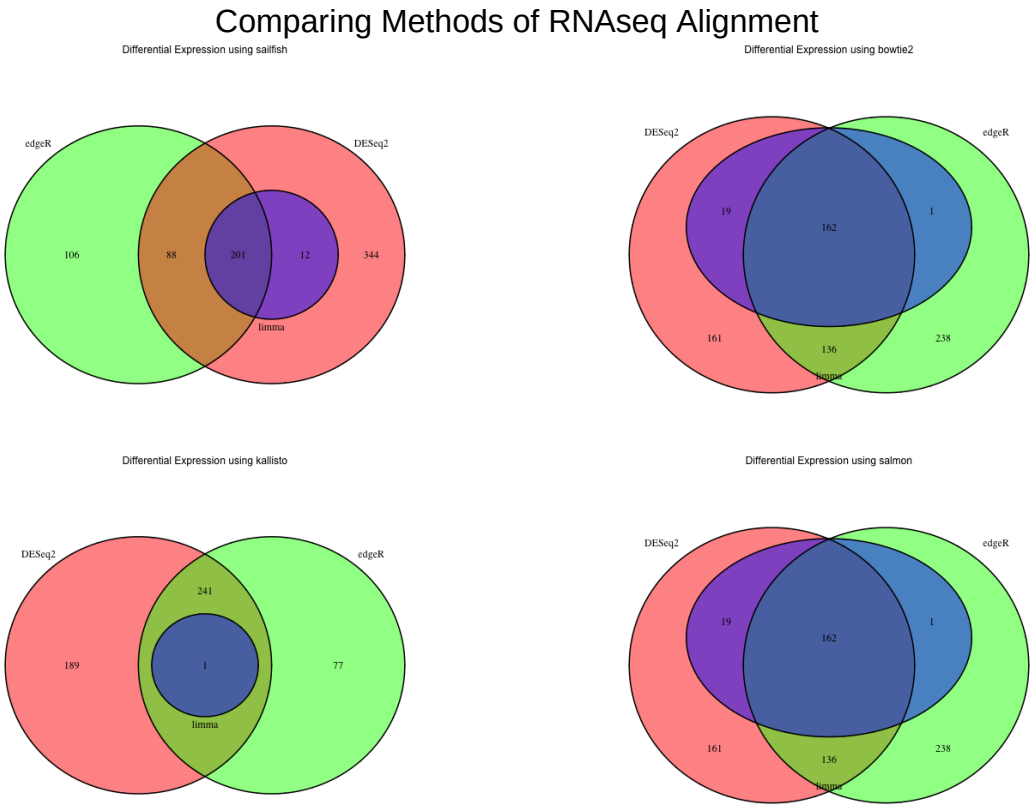
```
## quartz_off_screen
##                 3
```

```
dev.off()
```

```
## pdf
##   2
```

```
#saved plots as png's in the code chunks they were created in

aligners_venn_read <- lapply(list("venn_figs/sailfishvenn.png", "venn_figs/bowtievenn.png", "venn_figs/l
combo_alignersvenn <- lapply(aligners_venn_read, grid::rasterGrob) #creates a list of the read png's an
grid.arrange(grobs=combo_alignersvenn, ncol=2,
             top="Comparing Methods of RNAseq Alignment")
```




```
#saves plot as png
dev.copy(png,'venn_figs/combo_aligner_venn.png')
```

```
## quartz_off_screen
##                 3
```

```
dev.off()
```

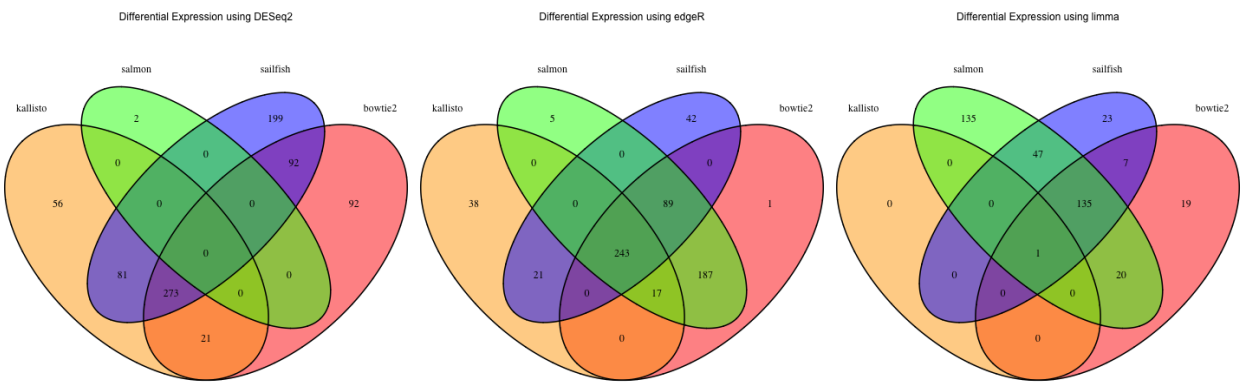
```
## pdf
##   2
```

Alligner Venn Diagrams Results:

Bowtie2 and Salmon have an identical distribution of genes picked up by each DE program. With kallisto, Limma only picked up one differentially expressed gene. For sailfish and kallisto, DESeq2 picked up the most differentially expressed genes. For Bowtie2 and Salmon, EdgeR picked up the most differentially expressed genes. Bowtie2 and Salmon had far more genes picked up by all three expression programs than kallisto and sailfish.

```
#saved plots as png's in the code chunks they were created in
de_venn_read <- lapply(list("venn_figs/deseq2venn.png", "venn_figs/edgrvenn.png", "venn_figs/limmavenn.
combo_devenn <- lapply(de_venn_read, grid::rasterGrob) #creates a list of the read png's and creates gr
```

```
grid.arrange(grobs=combo_devenn, ncol=3,
             top="Comparison of Differential Expression Analysis Packages")
```




```
#saves plot as png
dev.copy(png,'venn_figs/combo_de_venn.png')
```

```
## quartz_off_screen
##                 3
```

```
dev.off()
```

```
## pdf
##   2
```

Differential Expression Venn Diagrams Results:

Limma was the most conservative with the least total differentially expressed genes detected, DESeq2 was next most conservative, and EdgeR was least conservative. Limma only picked up a single differentially expressed gene with kallisto. DESeq2 only picked up two differentially expressed genes with Salmon. Only EdgeR picked up a significant amount of common genes for all four alligners.

Everything below here is a failure

```
DESeq_venn <- filter(complete_table, DE == "DESeq2")

nrow(filter(DESeq_venn, Align == "bowtie2")) #478
```

```
## [1] 478
```

```
nrow(filter(DESeq_venn, Align == "kallisto")) #431
```

```
## [1] 431
```

```
nrow(filter(DESeq_venn, Align == "salmon")) #2
```

```
## [1] 2
```

```
nrow(filter(DESeq_venn, Align == "sailfish")) #645
```

```
## [1] 645
```

```
#again, double checking that these add up to the total number of roles in the complete table
nrow(subset(DESeq_venn, Align == "bowtie2")) +
nrow(subset(DESeq_venn, Align == "kallisto")) +
nrow(subset(DESeq_venn, Align == "salmon")) +
nrow(subset(DESeq_venn, Align == "sailfish")) == nrow(DESeq_venn)
```

```
## [1] TRUE
```

```
#looking for overlap

#number of genes that Bowtie and kallisto share
combo_bowtie2_kallisto <- filter(DESeq_venn, Align == "bowtie2" | Align == "kallisto")
nrow(combo_bowtie2_kallisto) - n_distinct(combo_bowtie2_kallisto$GeneName) #294
```

```
## [1] 294
```

```
#bowtie2 and salmon
combo_bowtie2_salmon <- filter(DESeq_venn, Align == "bowtie2" | Align == "salmon")
nrow(combo_bowtie2_salmon) - n_distinct(combo_bowtie2_salmon$GeneName) #0
```

```
## [1] 0
```

```
#bowtie2 and sailfish
combo_bowtie2_sailfish <- filter(DESeq_venn, Align == "bowtie2" | Align == "sailfish")
nrow(combo_bowtie2_sailfish) - n_distinct(combo_bowtie2_sailfish$GeneName) #365
```

```
## [1] 365
```

```
#kallisto and salmon
combo_kallisto_salmon <- filter(DESeq_venn, Align == "kallisto" | Align == "salmon")
nrow(combo_kallisto_salmon) - n_distinct(combo_kallisto_salmon$GeneName) #0
```

```
## [1] 0
```

```
#kallisto and sailfish
combo_kallisto_sailfish <- filter(DESeq_venn, Align == "kallisto" | Align == "sailfish")
nrow(combo_kallisto_sailfish) - n_distinct(combo_kallisto_sailfish$GeneName) #354
```

```
## [1] 354
```

```
#salmon and sailfish
combo_salmon_sailfish <- filter(DESeq_venn, Align == "salmon" | Align == "sailfish")
nrow(combo_salmon_sailfish) - n_distinct(combo_salmon_sailfish$GeneName) #0
```

```
## [1] 0
```

```
#bowtie2, kallisto, salmon (123) =0


#bowtie2, kallisto, sailfish (124)


#bowtie2, salmon, sailfish (134) =0
```

```
#kallisto, salmon, sailfish (234) =0
#bowtie2 kallisto, salmon, sailfish 1234 =0
```

bowtie2= area1 kallisto= area2 salmon = area3 sailfish = area4

```
#grid.newpage()
#draw.quad.venn(area1 = 478, area2 = 431, area3 = 2, area4 = 645, n12=294, n13=0, n14=365, n23=0, n24=3
```

url to package info: https://cran.r-project.org/web/packages/VennDiagram/VennDiagram.pdf

url to tutorial: https://rstudio-pubs-static.s3.amazonaws.com/13301_6641d73cfac741a59c0a851feb99e98b.html

Looks like we can use "draw.triple.venn()" and "draw.quad.venn()". Check out the tutorial for some good info about specifics but it looks like we just need the sizes of each group (genes by bowtie, kallisto, etc). There are also some functions that appear to generate lists of different portions shared by the groups..I think the calculate.overlap() will do this and we just have to specify which groups you want to compare

Trying overlap function.

```
overlap3 <- calculate.overlap(x = list("bowtie2" = filter(DESeq_venn, Align == "bowtie2")$GeneName,"kal
overlap3
```

```
## $a6
## character(0)
##
## $a12
## character(0)
##
## $a11
##   [1] "FBtr0070392" "FBtr0071004" "FBtr0071849" "FBtr0072102" "FBtr0072506"
##   [6] "FBtr0072635" "FBtr0072642" "FBtr0072712" "FBtr0072957" "FBtr0073422"
##  [11] "FBtr0073426" "FBtr0073839" "FBtr0073974" "FBtr0075541" "FBtr0075558"
##  [16] "FBtr0075699" "FBtr0076454" "FBtr0076496" "FBtr0076500" "FBtr0076545"
##  [21] "FBtr0076668" "FBtr0076772" "FBtr0076935" "FBtr0077247" "FBtr0077644"
##  [26] "FBtr0078069" "FBtr0078136" "FBtr0078841" "FBtr0079199" "FBtr0080592"
##  [31] "FBtr0081614" "FBtr0081619" "FBtr0082889" "FBtr0083467" "FBtr0085105"
##  [36] "FBtr0085155" "FBtr0085307" "FBtr0085987" "FBtr0086280" "FBtr0086474"
##  [41] "FBtr0086506" "FBtr0086555" "FBtr0086613" "FBtr0087258" "FBtr0087617"
##  [46] "FBtr0087820" "FBtr0088052" "FBtr0088136" "FBtr0088266" "FBtr0088663"
##  [51] "FBtr0089263" "FBtr0089349" "FBtr0089581" "FBtr0089716" "FBtr0100575"
##  [56] "FBtr0110876" "FBtr0112735" "FBtr0112808" "FBtr0112994" "FBtr0113323"
##  [61] "FBtr0113380" "FBtr0114544" "FBtr0290078" "FBtr0299564" "FBtr0299837"
##  [66] "FBtr0299920" "FBtr0300889" "FBtr0301143" "FBtr0301492" "FBtr0301501"
##  [71] "FBtr0301850" "FBtr0302034" "FBtr0302291" "FBtr0302548" "FBtr0302561"
##  [76] "FBtr0302712" "FBtr0304802" "FBtr0304914" "FBtr0305063" "FBtr0305073"
##  [81] "FBtr0306087" "FBtr0306239" "FBtr0308621" "FBtr0308823" "FBtr0310064"
##  [86] "FBtr0310643" "FBtr0310669" "FBtr0330306" "FBtr0332038" "FBtr0332202"
##  [91] "FBtr0332403" "FBtr0332604" "FBtr0332911" "FBtr0332963" "FBtr0333071"
##  [96] "FBtr0333073" "FBtr0333326" "FBtr0334404" "FBtr0334883" "FBtr0335180"
## [101] "FBtr0336667" "FBtr0337020" "FBtr0339629" "FBtr0340135" "FBtr0342607"
## [106] "FBtr0342810" "FBtr0344989" "FBtr0345070" "FBtr0345429" "FBtr0346627"
## [111] "FBtr0347200" "FBtr0347567" "FBtr0472696" "FBtr0070393" "FBtr0070985"
## [116] "FBtr0071240" "FBtr0071835" "FBtr0071934" "FBtr0072087" "FBtr0072393"
## [121] "FBtr0072561" "FBtr0072956" "FBtr0072967" "FBtr0073269" "FBtr0073423"
## [126] "FBtr0073531" "FBtr0075039" "FBtr0075077" "FBtr0075204" "FBtr0075354"
## [131] "FBtr0075753" "FBtr0075811" "FBtr0076459" "FBtr0076541" "FBtr0076771"
```

```
## [136] "FBtr0077234" "FBtr0077560" "FBtr0077645" "FBtr0077972" "FBtr0078083"
## [141] "FBtr0078144" "FBtr0078191" "FBtr0078811" "FBtr0078937" "FBtr0079077"
## [146] "FBtr0079708" "FBtr0080565" "FBtr0080838" "FBtr0081004" "FBtr0081008"
## [151] "FBtr0081354" "FBtr0081484" "FBtr0082139" "FBtr0082158" "FBtr0082183"
## [156] "FBtr0083764" "FBtr0083967" "FBtr0084549" "FBtr0085094" "FBtr0085156"
## [161] "FBtr0085209" "FBtr0085423" "FBtr0085735" "FBtr0085755" "FBtr0086272"
## [166] "FBtr0086670" "FBtr0087004" "FBtr0087115" "FBtr0087295" "FBtr0087307"
## [171] "FBtr0087390" "FBtr0087533" "FBtr0087621" "FBtr0087789" "FBtr0088774"
## [176] "FBtr0088996" "FBtr0089348" "FBtr0089359" "FBtr0089361" "FBtr0089484"
## [181] "FBtr0089937" "FBtr0089963" "FBtr0100375" "FBtr0100457" "FBtr0100595"
## [186] "FBtr0110971" "FBtr0111030" "FBtr0112532" "FBtr0112851" "FBtr0113011"
## [191] "FBtr0113190" "FBtr0113470" "FBtr0273370" "FBtr0289981" "FBtr0290019"
## [196] "FBtr0290025" "FBtr0290076" "FBtr0290077" "FBtr0290327" "FBtr0299526"
## [201] "FBtr0299968" "FBtr0300190" "FBtr0300599" "FBtr0301403" "FBtr0302323"
## [206] "FBtr0302498" "FBtr0302592" "FBtr0302615" "FBtr0302645" "FBtr0302692"
## [211] "FBtr0302694" "FBtr0302713" "FBtr0302896" "FBtr0302938" "FBtr0303384"
## [216] "FBtr0303387" "FBtr0303389" "FBtr0303756" "FBtr0303787" "FBtr0303857"
## [221] "FBtr0303994" "FBtr0304621" "FBtr0304993" "FBtr0304997" "FBtr0306107"
## [226] "FBtr0306244" "FBtr0306930" "FBtr0307016" "FBtr0307057" "FBtr0307289"
## [231] "FBtr0307520" "FBtr0308065" "FBtr0308631" "FBtr0309828" "FBtr0310034"
## [236] "FBtr0310055" "FBtr0310647" "FBtr0310653" "FBtr0310661" "FBtr0329955"
## [241] "FBtr0329957" "FBtr0330118" "FBtr0330159" "FBtr0330725" "FBtr0330730"
## [246] "FBtr0332042" "FBtr0332057" "FBtr0332203" "FBtr0332260" "FBtr0332405"
## [251] "FBtr0332500" "FBtr0332503" "FBtr0332930" "FBtr0333020" "FBtr0333625"
## [256] "FBtr0333662" "FBtr0334282" "FBtr0334328" "FBtr0334867" "FBtr0335013"
## [261] "FBtr0335024" "FBtr0336616" "FBtr0336736" "FBtr0336759" "FBtr0339081"
## [266] "FBtr0339643" "FBtr0340709" "FBtr0342815" "FBtr0343358" "FBtr0343424"
## [271] "FBtr0344179" "FBtr0344980" "FBtr0345142"
##
## $a5
## character(0)
##
## $a7
## character(0)
##
## $a15
##  [1] "FBtr0070724" "FBtr0080886" "FBtr0300017" "FBtr0304005" "FBtr0332478"
##  [6] "FBtr0335278" "FBtr0339741" "FBtr0346714" "FBtr0073339" "FBtr0077409"
## [11] "FBtr0078204" "FBtr0079042" "FBtr0080514" "FBtr0081958" "FBtr0084659"
## [16] "FBtr0089343" "FBtr0302695" "FBtr0304933" "FBtr0307374" "FBtr0310016"
## [21] "FBtr0332116"
##
## $a4
## character(0)
##
## $a10
##  [1] "FBtr0071246" "FBtr0071599" "FBtr0071613" "FBtr0073120" "FBtr0075554"
##  [6] "FBtr0075928" "FBtr0076364" "FBtr0076453" "FBtr0078346" "FBtr0080942"
## [11] "FBtr0081257" "FBtr0081353" "FBtr0082017" "FBtr0084868" "FBtr0085000"
## [16] "FBtr0085252" "FBtr0087090" "FBtr0087369" "FBtr0088399" "FBtr0091719"
## [21] "FBtr0113386" "FBtr0114584" "FBtr0273416" "FBtr0301212" "FBtr0302453"
## [26] "FBtr0303046" "FBtr0308564" "FBtr0310673" "FBtr0330145" "FBtr0332125"
## [31] "FBtr0332720" "FBtr0333058" "FBtr0333659" "FBtr0333852" "FBtr0334078"
## [36] "FBtr0334633" "FBtr0339391" "FBtr0339682" "FBtr0339696" "FBtr0340190"
```

```
## [41] "FBtr0342806" "FBtr0342813" "FBtr0342814" "FBtr0342818" "FBtr0344978"
## [46] "FBtr0346541" "FBtr0346713" "FBtr0347568" "FBtr0072410" "FBtr0072864"
## [51] "FBtr0073421" "FBtr0073527" "FBtr0074337" "FBtr0074916" "FBtr0076218"
## [56] "FBtr0079648" "FBtr0079667" "FBtr0080589" "FBtr0080932" "FBtr0081211"
## [61] "FBtr0082812" "FBtr0084054" "FBtr0084447" "FBtr0084958" "FBtr0084963"
## [66] "FBtr0085377" "FBtr0085997" "FBtr0088224" "FBtr0088337" "FBtr0088370"
## [71] "FBtr0088902" "FBtr0089877" "FBtr0111028" "FBtr0113291" "FBtr0113451"
## [76] "FBtr0273385" "FBtr0301131" "FBtr0301915" "FBtr0305218" "FBtr0308633"
## [81] "FBtr0310438" "FBtr0330263" "FBtr0330722" "FBtr0331931" "FBtr0333671"
## [86] "FBtr0334327" "FBtr0334553" "FBtr0334762" "FBtr0339234" "FBtr0340289"
## [91] "FBtr0342811" "FBtr0345487"
##
## $a13
## character(0)
##
## $a8
##  [1] "FBtr0084465" "FBtr0300272" "FBtr0332167" "FBtr0273275" "FBtr0302977"
##  [6] "FBtr0080839" "FBtr0301824" "FBtr0070458" "FBtr0339428" "FBtr0303709"
## [11] "FBtr0100470" "FBtr0073619" "FBtr0331663" "FBtr0330651" "FBtr0301884"
## [16] "FBtr0339563" "FBtr0089091" "FBtr0340405" "FBtr0334846" "FBtr0335475"
## [21] "FBtr0100298" "FBtr0301708" "FBtr0303385" "FBtr0336676" "FBtr0087493"
## [26] "FBtr0330225" "FBtr0334405" "FBtr0089516" "FBtr0085001" "FBtr0111065"
## [31] "FBtr0072767" "FBtr0085899" "FBtr0087089" "FBtr0302905" "FBtr0333657"
## [36] "FBtr0339974" "FBtr0087870" "FBtr0339418" "FBtr0340125" "FBtr0344277"
## [41] "FBtr0100643" "FBtr0087308" "FBtr0336467" "FBtr0305668" "FBtr0308638"
## [46] "FBtr0334585" "FBtr0301450" "FBtr0070457" "FBtr0075927" "FBtr0081504"
## [51] "FBtr0087257" "FBtr0345896" "FBtr0330269" "FBtr0332529" "FBtr0081352"
## [56] "FBtr0304995" "FBtr0074276" "FBtr0334849" "FBtr0082094" "FBtr0308201"
## [61] "FBtr0339189" "FBtr0333595" "FBtr0331642" "FBtr0070965" "FBtr0336904"
## [66] "FBtr0306616" "FBtr0339144" "FBtr0089649" "FBtr0332046" "FBtr0088521"
## [71] "FBtr0332142" "FBtr0336620" "FBtr0300039" "FBtr0344703" "FBtr0339493"
## [76] "FBtr0076135" "FBtr0078430" "FBtr0302322" "FBtr0303749" "FBtr0335521"
## [81] "FBtr0335012"
##
## $a2
## character(0)
##
## $a9
##  [1] FBtr0070241 FBtr0072993 FBtr0072994 FBtr0073492 FBtr0074528
##  [6] FBtr0075199 FBtr0076797 FBtr0077236 FBtr0080247 FBtr0081686
## [11] FBtr0082069 FBtr0082093 FBtr0085137 FBtr0088363 FBtr0088571
## [16] FBtr0088699 FBtr0100582 FBtr0110823 FBtr0112807 FBtr0289940
## [21] FBtr0290279 FBtr0300192 FBtr0300245 FBtr0300337 FBtr0301622
## [26] FBtr0304925 FBtr0307092 FBtr0308213 FBtr0308594 FBtr0310435
## [31] FBtr0329879 FBtr0330617 FBtr0330631 FBtr0331308 FBtr0332912
## [36] FBtr0332960 FBtr0333860 FBtr0334778 FBtr0335414 FBtr0336942
## [41] FBtr0339443 FBtr0340240 FBtr0342893 FBtr0343527 FBtr0344860
## [46] FBtr0346957 FBtr0074842 FBtr0075560 FBtr0077259 FBtr0077910
## [51] FBtr0078687 FBtr0078812 FBtr0080261 FBtr0082117 FBtr0082494
## [56] FBtr0083058 FBtr0085146 FBtr0085353 FBtr0085708 FBtr0086983
## [61] FBtr0087528 FBtr0088100 FBtr0088522 FBtr0088602 FBtr0089162
## [66] FBtr0100129 FBtr0110938 FBtr0112740 FBtr0113113 FBtr0301537
## [71] FBtr0301651 FBtr0302691 FBtr0303790 FBtr0304683 FBtr0308828
## [76] FBtr0309256 FBtr0310660 FBtr0321321 FBtr0329839 FBtr0330054
```

```
## [81] FBtr0331952 FBtr0332122 FBtr0334067 FBtr0335206 FBtr0337056
## [86] FBtr0339407 FBtr0339593 FBtr0343095 FBtr0343504 FBtr0345204
## [91] FBtr0346632 FBtr0347553
## 1008 Levels: FBtr0070241 FBtr0070392 FBtr0070393 FBtr0070724 ... FBtr0446106
##
## $a14
##  [1] FBtr0305911 FBtr0071902 FBtr0344118 FBtr0074800 FBtr0345108
##  [6] FBtr0310658 FBtr0309504 FBtr0273423 FBtr0084282 FBtr0334322
## [11] FBtr0333545 FBtr0071537 FBtr0334924 FBtr0310266 FBtr0304794
## [16] FBtr0083122 FBtr0340347 FBtr0302833 FBtr0308834 FBtr0081992
## [21] FBtr0302355 FBtr0077349 FBtr0083904 FBtr0087147 FBtr0080892
## [26] FBtr0333972 FBtr0340212 FBtr0076678 FBtr0346946 FBtr0305679
## [31] FBtr0088549 FBtr0306917 FBtr0088261 FBtr0089409 FBtr0306520
## [36] FBtr0332851 FBtr0113140 FBtr0301796 FBtr0087415 FBtr0089875
## [41] FBtr0086503 FBtr0086055 FBtr0330330 FBtr0306919 FBtr0305968
## [46] FBtr0330048 FBtr0309099 FBtr0072740 FBtr0303912 FBtr0300877
## [51] FBtr0332205 FBtr0345034 FBtr0301857 FBtr0290197 FBtr0086255
## [56] FBtr0329895
## 1008 Levels: FBtr0070241 FBtr0070392 FBtr0070393 FBtr0070724 ... FBtr0446106
##
## $a1
## [1] FBtr0332407 FBtr0273189
## 1008 Levels: FBtr0070241 FBtr0070392 FBtr0070393 FBtr0070724 ... FBtr0446106
##
## $a3
##   [1] FBtr0304990 FBtr0305263 FBtr0080171 FBtr0082992 FBtr0083552
##   [6] FBtr0082522 FBtr0084911 FBtr0342890 FBtr0344288 FBtr0333059
##  [11] FBtr0331844 FBtr0334147 FBtr0089486 FBtr0089487 FBtr0081362
##  [16] FBtr0336697 FBtr0301964 FBtr0334561 FBtr0113200 FBtr0086170
##  [21] FBtr0329949 FBtr0335183 FBtr0330176 FBtr0308644 FBtr0071535
##  [26] FBtr0330729 FBtr0304613 FBtr0110921 FBtr0344560 FBtr0344873
##  [31] FBtr0089934 FBtr0343189 FBtr0333334 FBtr0304665 FBtr0344608
##  [36] FBtr0331664 FBtr0073425 FBtr0073292 FBtr0301487 FBtr0300649
##  [41] FBtr0310621 FBtr0079061 FBtr0343016 FBtr0302091 FBtr0342805
##  [46] FBtr0300815 FBtr0080263 FBtr0301851 FBtr0088895 FBtr0337016
##  [51] FBtr0090014 FBtr0077828 FBtr0333726 FBtr0332581 FBtr0075264
##  [56] FBtr0339825 FBtr0333219 FBtr0070966 FBtr0302577 FBtr0076263
##  [61] FBtr0074872 FBtr0344790 FBtr0335263 FBtr0310668 FBtr0347024
##  [66] FBtr0074777 FBtr0112706 FBtr0345069 FBtr0113129 FBtr0077479
##  [71] FBtr0290026 FBtr0333043 FBtr0332003 FBtr0339959 FBtr0301372
##  [76] FBtr0343342 FBtr0339520 FBtr0080382 FBtr0111054 FBtr0084601
##  [81] FBtr0309096 FBtr0303148 FBtr0273412 FBtr0077831 FBtr0074745
##  [86] FBtr0100461 FBtr0332847 FBtr0303750 FBtr0330679 FBtr0333013
##  [91] FBtr0332836 FBtr0332589 FBtr0336659 FBtr0344967 FBtr0082057
##  [96] FBtr0343517 FBtr0306576 FBtr0301337 FBtr0332862 FBtr0343633
## [101] FBtr0332726 FBtr0344824 FBtr0078384 FBtr0112501 FBtr0308827
## [106] FBtr0290281 FBtr0085126 FBtr0301000 FBtr0334279 FBtr0290320
## [111] FBtr0112885 FBtr0072722 FBtr0333119 FBtr0073187 FBtr0083252
## [116] FBtr0112920 FBtr0086530 FBtr0344438 FBtr0083613 FBtr0345036
## [121] FBtr0308034 FBtr0100352 FBtr0304053 FBtr0070725 FBtr0306255
## [126] FBtr0085323 FBtr0075297 FBtr0111029 FBtr0081527 FBtr0344230
## [131] FBtr0334309 FBtr0344403 FBtr0112878 FBtr0446106 FBtr0339755
## [136] FBtr0344439 FBtr0343418 FBtr0300265 FBtr0084212 FBtr0345071
## [141] FBtr0330331 FBtr0073615 FBtr0100656 FBtr0074715 FBtr0333912
```

```
## [146] FBtr0299830 FBtr0088342 FBtr0273230 FBtr0073393 FBtr0303445
## [151] FBtr0301766 FBtr0343050 FBtr0076417 FBtr0302732 FBtr0345249
## [156] FBtr0077707 FBtr0090028 FBtr0300415 FBtr0332089 FBtr0075266
## [161] FBtr0087930 FBtr0100449 FBtr0081213 FBtr0306233 FBtr0332999
## [166] FBtr0308642 FBtr0082615 FBtr0079148 FBtr0303205 FBtr0307900
## [171] FBtr0077235 FBtr0070265 FBtr0331683 FBtr0330019 FBtr0113357
## [176] FBtr0332317 FBtr0075173 FBtr0301278 FBtr0306657 FBtr0100652
## [181] FBtr0306158 FBtr0302297 FBtr0084229 FBtr0078389 FBtr0344353
## [186] FBtr0074432 FBtr0076250 FBtr0334011 FBtr0344549 FBtr0308627
## [191] FBtr0086617 FBtr0086926 FBtr0344847 FBtr0301500 FBtr0343580
## [196] FBtr0070284 FBtr0344861 FBtr0299967 FBtr0077466
## 1008 Levels: FBtr0070241 FBtr0070392 FBtr0070393 FBtr0070724 ... FBtr0446106
```

```
#a6=0
#a12=0
#a11=273
#a5 = 0
#a7=0
#a19=21
#a4=0
#a10=92
#a13=0
#a8=81
#a2=0
#a9=92
#a14=56
#a1=2
#a3=199
```

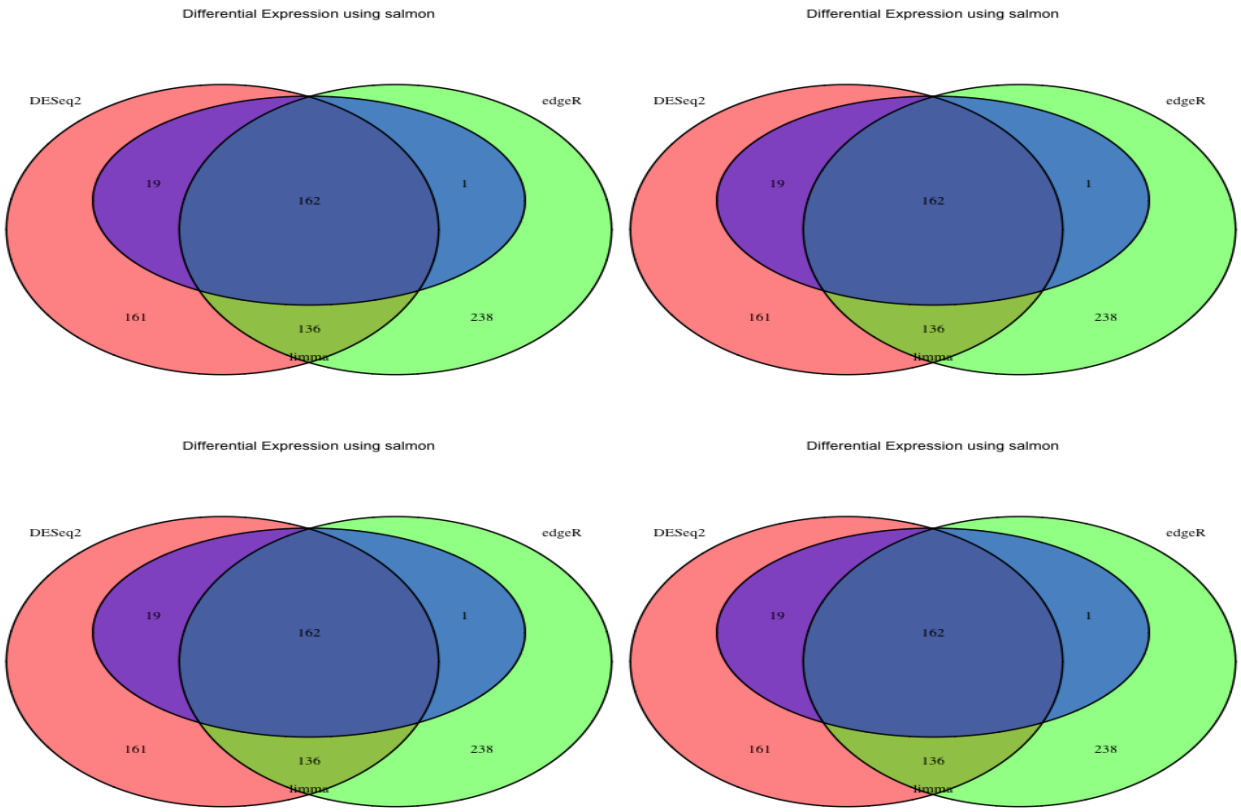
```
library("png") # for reading in PNGs

# Reading png's
sfv <- readPNG("venn_figs/sailfishvenn.png")
btv <- readPNG("venn_figs/bowtievenn.png")
kv <- readPNG("venn_figs/kallistovenn.png")
sv <- readPNG("venn_figs/salmonvenn.png")

# setup plot
#dev.off()
par(mai=rep(0,4)) # no margins

# layout the plots into a matrix w/ 2 columns, by row
layout(matrix(1:4, ncol=2, byrow=TRUE))

# do the plotting
for(i in 1:4) {
  plot(NA,xlim=0:1,ylim=0:1,bty="n",axes=0,xaxs = 'i',yaxs='i')
  rasterImage(sfv,0,0,1,1)
  rasterImage(btv,0,0,1,1)
  rasterImage(kv,0,0,1,1)
  rasterImage(sv,0,0,1,1)
}
```




```
# write to PDF
#dev.print(pdf, "output.pdf")
```

“‘ Sources:

combo images https://cran.r-project.org/web/packages/gridExtra/vignettes/arrangeGrob.html https://www.rdocumentation.org/packages/grid/versions/3.5.3 https://stackoverflow.com/questions/23750305/how-can-i-make-multipanel-plots-of-several-png-files-in-r?lq=1