Grape Differential Expression Analysis with EdgeR

Scott Teresi

Purpose:

Produce a matrix of differentially expressed genes for the grape RNA data, a smear plot, and a summary table. I will be using the package edgeR for this task.

Accessibility and Help:

The following guide, source code, and other components of the grape expression analysis pipeline can be found at Grape_RNA_Seq_Expression_Analysis Github page. This page includes general information, the keys of the samples, and the list of comparisons desired.

Installation and Loading of Library:

Using this link as reference, please install **edgeR**. I have set the chunk evaluation clause to FALSE, as you only need to install once and should do it manually.

```
if (!requireNamespace("BiocManager", quietly = TRUE))
    install.packages("BiocManager")

BiocManager::install("edgeR")
```

We also need tidyverse which includes dplyr but just to be safe we will install both with:

```
install.packages("tidyverse")
install.packages("dplyr")
```

And then we will load the libraries with:

```
library(edgeR)
library(tidyverse)
library(dplyr)
```

Loading the Data Into R (Part 1):

Here we will load all of our data from HTSeq into R. There are 3 files. Two of which, the 802 and 724 data sets can be lumped together as they have similar samples. The 809 data set belongs on its own. Below I import the data, perform the appropriate merges and reorient the data. I also remove rows that are completely 0s, because they are uninformative rows and will cause statistical issues. This is a common step.

```
# Change this to your own path
setwd('/home/scott/Documents/Uni/Research/Projects/Grape_RNA')

# Import data
Seq_809 = read.csv('809_Seq.tsv', sep='\t',
           stringsAsFactors = FALSE,
           header = TRUE)

Seq_802 = read.csv('802_Seq.tsv', sep='\t',
           stringsAsFactors = FALSE,
           header = TRUE)

Seq_724 = read.csv('724_Seq.tsv', sep='\t',
           stringsAsFactors = FALSE,
           header = TRUE)

# Merge that data
input_data_724_802 = merge(Seq_802, Seq_724, by = 'gene_id')
# The other standalone is
input_data_809 = Seq_809

# Remove those extraneous rows from HTSeq
input_data_724_802 = tail(input_data_724_802, -5)
input_data_809 = tail(input_data_809, -5)

# Set the index
rownames(input_data_724_802) = input_data_724_802$gene_id
rownames(input_data_809) = input_data_809$gene_id

# Get rid of gene name column because it is now the index
input_data_724_802 = subset(input_data_724_802, select = -c(gene_id))
input_data_809 = subset(input_data_809, select = -c(gene_id))

# Remove the rows that are completely 0s, uninformative rows
input_data_724_802 = input_data_724_802[apply(input_data_724_802, 1, function(x) {
  !all(x == 0)}), ]
input_data_809 = input_data_809[apply(input_data_809, 1, function(x) {
```

```
  !all(x == 0)}), ]

# Remove the data objects to clear the clutter
rm(Seq_724, Seq_802, Seq_809)
```

Loading the Data Into R (Part 2):

Here we will filter the data and add the appropriate "metadata" so that we can easily recognie each sample. I write these as a function so that we can easily utilize them later inside the individual sample comparison chunks.

```
# Rules for the 724 & and 802 Groupings
Filter_724_802 = function(input_data_724_802) {
  my_data = input_data_724_802
  # Add two empty rows at the end of the data frame to be filled with the
  # experimental factors that we later plug in.
  my_data[(nrow(my_data) + 1):(nrow(my_data) + 2), ] = NA

  # Loop through the columns and assign experimental factors based on the
  # sample names, filling the last two rows
  columns = colnames(my_data)

  for (i in 1:ncol(my_data)) {
    if (grepl("W_C_*", colnames(my_data)[i])) {
      my_data[(nrow(my_data) - 1),i] <- "Water"
      my_data[(nrow(my_data)), i] <- "Control"

      # GA and CK
    } else if (grepl("GA3_CK_C_*", colnames(my_data)[i])) {
      my_data[(nrow(my_data) - 1),i] <- "GA_CK"
      my_data[(nrow(my_data)), i] <- "Control"
    } else if (grepl("GA3_[^CK_C]", colnames(my_data)[i])) {
      my_data[(nrow(my_data) - 1),i] <- "GA"
      my_data[(nrow(my_data)), i] <- "Treatment"

      # AUX
    } else if (grepl("AUX_C_", colnames(my_data)[i])) {
      my_data[(nrow(my_data) - 1),i] <- "AUX"
      my_data[(nrow(my_data)), i] <- "Control"
    } else if (grepl("AUX_[^C]", colnames(my_data)[i])) {
      my_data[(nrow(my_data) - 1),i] <- "AUX"
      my_data[(nrow(my_data)), i] <- "Treatment"
```

```
      # CK
    } else if (grepl("^CK_", colnames(my_data)[i])) {
      my_data[(nrow(my_data) - 1),i] <- "CK"
      my_data[(nrow(my_data)), i] <- "Treatment"
    }
  }
# Update rows with "metadata" on each sample's identity
  row.names(my_data)[(nrow(my_data) - 1) : (nrow(my_data))] = c("Chemical", "Treatment")
  Filtered_724_802 = my_data
  rm(my_data)
  return(Filtered_724_802)
}

Filter_809 = function(input_data_809) {
  my_data = input_data_809
  # Add three empty rows at the end of the data frame to be filled with the
    # experimental factors that we later plug in.
  my_data[(nrow(my_data) + 1):(nrow(my_data) + 3), ] = NA

  # Loop through the columns and assign experimental factors based on the
  # sample names, filling the last two rows
  columns = colnames(my_data)

  for (i in 1:ncol(my_data)) {
    # SB
    if (grepl("SB\\.C[^K](.*?)_S", colnames(my_data)[i])) {
    my_data[(nrow(my_data) - 2),i] <- "Untransformed"
    my_data[(nrow(my_data) - 1), i] <- "Control_Region"
    my_data[(nrow(my_data)), i] <- "No_Gall"
  } else if (grepl("SB\\.CK(.*?)G", colnames(my_data)[i])) {
    my_data[(nrow(my_data) - 2),i] <- "Untransformed"
    my_data[(nrow(my_data) - 1), i] <- "Control_Region"
    my_data[(nrow(my_data)), i] <- "Gall"

    # DPR
  } else if (grepl("DPR(.*?)G_S", colnames(my_data)[i])) {
    my_data[(nrow(my_data) - 2),i] <- "Untransformed"
    my_data[(nrow(my_data) - 1), i] <- "Empty_Vector"
    my_data[(nrow(my_data)), i] <- "Gall"
  } else if (grepl("DPR(.*?)[^G]_S*", colnames(my_data)[i])) {
    my_data[(nrow(my_data) - 2),i] <- "Untransformed"
    my_data[(nrow(my_data) - 1), i] <- "Empty_Vector"
    my_data[(nrow(my_data)), i] <- "No_Gall"
```

```
    # WUS Reg 1
  } else if (grepl("X1(.*?)G_S", colnames(my_data)[i])) {
    my_data[(nrow(my_data) - 2),i] <- "Transformed"
    my_data[(nrow(my_data) - 1), i] <- "WUS_Reg_1"
    my_data[(nrow(my_data)), i] <- "Gall"
  } else if (grepl("X1(.*?)[^G]_S", colnames(my_data)[i])) {
    my_data[(nrow(my_data) - 2),i] <- "Transformed"
    my_data[(nrow(my_data) - 1), i] <- "WUS_Reg_1"
    my_data[(nrow(my_data)), i] <- "No_Gall"

    # WUS Reg 2
  } else if (grepl("X2(.*?)G_S", colnames(my_data)[i])) {
    my_data[(nrow(my_data) - 2),i] <- "Transformed"
    my_data[(nrow(my_data) - 1), i] <- "WUS_Reg_2"
    my_data[(nrow(my_data)), i] <- "Gall"
  } else if (grepl("X2(.*?)[^G]_S", colnames(my_data)[i])) {
    my_data[(nrow(my_data) - 2),i] <- "Transformed"
    my_data[(nrow(my_data) - 1), i] <- "WUS_Reg_2"
    my_data[(nrow(my_data)), i] <- "No_Gall"

    # LFY Reg 1
  } else if (grepl("X3(.*?)G_S", colnames(my_data)[i])) {
    my_data[(nrow(my_data) - 2),i] <- "Transformed"
    my_data[(nrow(my_data) - 1), i] <- "LFY_Reg_1"
    my_data[(nrow(my_data)), i] <- "Gall"
  } else if (grepl("X3(.*?)[^G]_S", colnames(my_data)[i])) {
    my_data[(nrow(my_data) - 2),i] <- "Transformed"
    my_data[(nrow(my_data) - 1), i] <- "LFY_Reg_1"
    my_data[(nrow(my_data)), i] <- "No_Gall"

    # LFY Reg 2
  } else if (grepl("X4(.*?)G_S", colnames(my_data)[i])) {
    my_data[(nrow(my_data) - 2),i] <- "Transformed"
    my_data[(nrow(my_data) - 1), i] <- "LFY_Reg_2"
    my_data[(nrow(my_data)), i] <- "Gall"
  } else if (grepl("X4(.*?)[^G]_S", colnames(my_data)[i])) {
    my_data[(nrow(my_data) - 2),i] <- "Transformed"
    my_data[(nrow(my_data) - 1), i] <- "LFY_Reg_2"
    my_data[(nrow(my_data)), i] <- "No_Gall"
  }
  }
# Update rows with "metadata" on each sample's identity
  row.names(my_data)[(nrow(my_data) - 2) : (nrow(my_data))] = c("Transformation", "Knock
  Filtered_809 = my_data
```

```
  rm(my_data)
  return(Filtered_809)
}
```

Project 1:

Here we are doing project 1, which includes the GA, CK, and AUX experiments ## GA Comparisons

```
Filtered_724_802 = Filter_724_802(input_data_724_802)
# Switch to output folder
setwd('/home/scott/Documents/Uni/Research/Projects/Grape_RNA/Output/Project_1')
#-----------------------------------
# Transformed vs. Untransformed
# GA3_CK vs GA3
# The order is control vs treatment
GA3_Comparisons_Function = function(Filtered_724_802) {
  # Subset the Data
  GA3_CK =  select(Filtered_724_802, matches("GA3_CK_C_*"))
  GA3 =  select(Filtered_724_802, matches("GA3_[^CK_C]"))
  GA3_Comparisons = list(GA3_CK, GA3)
  # Return list of subsetted data
  return(GA3_Comparisons)
}
GA3_Comparisons = GA3_Comparisons_Function(Filtered_724_802)
GA3_CK = GA3_Comparisons[[1]]
GA3 = GA3_Comparisons[[2]]

# Specify how to merge
Counts = merge(GA3_CK, GA3, by = 'row.names')
rm(GA3_CK, GA3)

# Make grouping for treatment groups.
rownames(Counts) <- Counts$Row.names
Counts = subset(Counts, select = -c(Row.names))
my_grouping = c(tail(Counts, 1)) # treatment vs control grouping declared
Counts = head(Counts, -1) # drop Treatment row
Counts = tail(Counts, -1) # drop Chemical row

Counts = as.matrix.data.frame(Counts)
x = rownames(Counts)
# We need the gene names for later
Gene_Row_Key = data.frame("Num"=rownames(as.data.frame(x)),"Gene"=x)
```

```r
rm(x)

Counts = apply(Counts, 2, as.numeric)
D = DGEList(Counts, group = unlist(my_grouping))
D = calcNormFactors(D)
D_Samples = D$samples

D = estimateCommonDisp(D)
D = estimateTagwiseDisp(D)
Fish_Exact = exactTest(D)
topTags = topTags(Fish_Exact)

# P = 0.05
simplified_DGE = decideTestsDGE(Fish_Exact, p=0.05, adjust="BH")
my_test = rownames(Fish_Exact)[as.logical(simplified_DGE)]
jpeg('GA3Ck_vs_GA3_Smear.png')
plotSmear(Fish_Exact, de.tags=my_test, main = "GA3Ck vs. GA3 Smear Plot")
abline(h=c(0,10))
dev.off()
```

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

```r
# Write summary table
write.table(summary(simplified_DGE), file = 'GA3CK_vs_GA3_Summary.txt', quote = F, sep =

# Write direction of differntial expression table
simplified_DGE_frame = data.frame(simplified_DGE)
colnames(simplified_DGE_frame) = 'Direction_Differentially_Regulated'
row.names(simplified_DGE_frame) = Gene_Row_Key$Gene
Trans_v_Untrans_Wus_SBC_Direction = as_tibble(rownames_to_column(simplified_DGE_frame,
write_tsv(Trans_v_Untrans_Wus_SBC_Direction, 'GA3CK_vs_GA3_Direction.tsv')

# Write full output
row.names(Fish_Exact) = Gene_Row_Key$Gene
Fish_tbl = as_tibble(rownames_to_column(Fish_Exact$table, var = 'Gene_Name'))
write_tsv(Fish_tbl, 'GA3CK_vs_GA3_Full.tsv')

# Clear workspace
rm(Gene_Row_Key, D, D_Samples, Fish_Exact, my_grouping, Counts, topTags, Fish_tbl, GA3_C
```

Cytokinin comparisons

```
Filtered_724_802 = Filter_724_802(input_data_724_802)
# Switch to output folder
setwd('/home/scott/Documents/Uni/Research/Projects/Grape_RNA/Output/Project_1')
#-----------------------------------
# Transformed vs. Untransformed
# GA3_CK vs GA3
# The order is control vs treatment
CK_Comparisons_Function = function(Filtered_724_802) {
  # Subset the Data
  GA3_CK =  select(Filtered_724_802, matches("GA3_CK_C_*"))
  CK =  select(Filtered_724_802, matches("^CK_"))
  CK_Comparisons = list(GA3_CK, CK)
  # Return list of subsetted data
  return(CK_Comparisons)
}
CK_Comparisons = CK_Comparisons_Function(Filtered_724_802)
GA3_CK = CK_Comparisons[[1]]
CK = CK_Comparisons[[2]]

# Specify how to merge
Counts = merge(GA3_CK, CK, by = 'row.names')
rm(GA3_CK, CK)

# Make grouping for treatment groups.
rownames(Counts) <- Counts$Row.names
Counts = subset(Counts, select = -c(Row.names))
my_grouping = c(tail(Counts, 1)) # treatment vs control grouping declared
Counts = head(Counts, -1) # drop Treatment row
Counts = tail(Counts, -1) # drop Chemical row

Counts = as.matrix.data.frame(Counts)
x = rownames(Counts)
# We need the gene names for later
Gene_Row_Key = data.frame("Num"=rownames(as.data.frame(x)),"Gene"=x)
rm(x)

Counts = apply(Counts, 2, as.numeric)
D = DGEList(Counts, group = unlist(my_grouping))
D = calcNormFactors(D)
D_Samples = D$samples

D = estimateCommonDisp(D)
```

```
D = estimateTagwiseDisp(D)
Fish_Exact = exactTest(D)
topTags = topTags(Fish_Exact)

# P = 0.05
simplified_DGE = decideTestsDGE(Fish_Exact, p=0.05, adjust="BH")
my_test = rownames(Fish_Exact)[as.logical(simplified_DGE)]
jpeg('GA3Ck_vs_CK_Smear.png')
plotSmear(Fish_Exact, de.tags=my_test, main = "GA3Ck vs. CK Smear Plot")
abline(h=c(0,10))
dev.off()
```

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

```
# Write summary table
write.table(summary(simplified_DGE), file = 'GA3CK_vs_CK_Summary.txt', quote = F, sep =

# Write direction of differntial expression table
simplified_DGE_frame = data.frame(simplified_DGE)
colnames(simplified_DGE_frame) = 'Direction_Differentially_Regulated'
row.names(simplified_DGE_frame) = Gene_Row_Key$Gene
Trans_v_Untrans_Wus_SBC_Direction = as_tibble(rownames_to_column(simplified_DGE_frame,
write_tsv(Trans_v_Untrans_Wus_SBC_Direction, 'GA3CK_vs_CK_Direction.tsv')

# Write full output
row.names(Fish_Exact) = Gene_Row_Key$Gene
Fish_tbl = as_tibble(rownames_to_column(Fish_Exact$table, var = 'Gene_Name'))
write_tsv(Fish_tbl, 'GA3CK_vs_CK_Full.tsv')

# Clear workspace
rm(Gene_Row_Key, D, D_Samples, Fish_Exact, my_grouping, Counts, topTags, Fish_tbl, CK_Co
```

Auxin Comparisons

```
Filtered_724_802 = Filter_724_802(input_data_724_802)
# Switch to output folder
setwd('/home/scott/Documents/Uni/Research/Projects/Grape_RNA/Output/Project_1')
#-----------------------------------
# Control vs. Treatment
# AUX_C vs AUX
AUX_Comparisons_Function = function(Filtered_724_802) {
  # Subset the Data
```

```
  AUX_C =  select(Filtered_724_802, matches("AUX_C_"))
  AUX_T =  select(Filtered_724_802, matches("AUX_[^C]"))
  AUX_Comparisons = list(AUX_C, AUX_T)
  # Return list of subsetted data
  return(AUX_Comparisons)
}
AUX_Comparisons = AUX_Comparisons_Function(Filtered_724_802)
AUX_C = AUX_Comparisons[[1]]
AUX_T = AUX_Comparisons[[2]]

# Specify how to merge
Counts = merge(AUX_C, AUX_T, by = 'row.names')
rm(AUX_C, AUX_T)

# Make grouping for treatment groups.
rownames(Counts) <- Counts$Row.names
Counts = subset(Counts, select = -c(Row.names))
my_grouping = c(tail(Counts, 1)) # treatment vs control grouping declared
Counts = head(Counts, -1) # drop Treatment row
Counts = tail(Counts, -1) # drop Chemical row

Counts = as.matrix.data.frame(Counts)
x = rownames(Counts)
# We need the gene names for later
Gene_Row_Key = data.frame("Num"=rownames(as.data.frame(x)),"Gene"=x)
rm(x)

Counts = apply(Counts, 2, as.numeric)
D = DGEList(Counts, group = unlist(my_grouping))
D = calcNormFactors(D)
D_Samples = D$samples

D = estimateCommonDisp(D)
D = estimateTagwiseDisp(D)
Fish_Exact = exactTest(D)
topTags = topTags(Fish_Exact)

# P = 0.05
simplified_DGE = decideTestsDGE(Fish_Exact, p=0.05, adjust="BH")
my_test = rownames(Fish_Exact)[as.logical(simplified_DGE)]
jpeg('AUX_C_vs_AUX_Smear.png')
plotSmear(Fish_Exact, de.tags=my_test, main = "Auxin Control vs. Auxin Treatment Smear P
abline(h=c(0,10))
dev.off()
```

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

```
# Write summary table
write.table(summary(simplified_DGE), file = 'AUX_C_vs_AUX_Summary.txt', quote = F, sep =

# Write direction of differntial expression table
simplified_DGE_frame = data.frame(simplified_DGE)
colnames(simplified_DGE_frame) = 'Direction_Differentially_Regulated'
row.names(simplified_DGE_frame) = Gene_Row_Key$Gene
Trans_v_Untrans_Wus_SBC_Direction = as_tibble(rownames_to_column(simplified_DGE_frame, 
write_tsv(Trans_v_Untrans_Wus_SBC_Direction, 'AUX_C_vs_AUX_Direction.tsv')

# Write full output
row.names(Fish_Exact) = Gene_Row_Key$Gene
Fish_tbl = as_tibble(rownames_to_column(Fish_Exact$table, var = 'Gene_Name'))
write_tsv(Fish_tbl, 'AUX_C_vs_AUX_Full.tsv')

# Clear workspace
rm(Gene_Row_Key, D, D_Samples, Fish_Exact, my_grouping, Counts, topTags, Fish_tbl, CK_Co
```

```
## Warning in rm(Gene_Row_Key, D, D_Samples, Fish_Exact, my_grouping,
## Counts, : object 'CK_Comparisons' not found
```

Project 2:

WUS1 vs SBC Comparisons

```
Filtered_809 = Filter_809(input_data_809)
# Switch to output folder
setwd('/home/scott/Documents/Uni/Research/Projects/Grape_RNA/Output/Project_2')
#-----------------------------------
WUS1_No_Gall_Comparisons_Function = function(Filtered_809) {
  # Subset the Data
  SB_C = select(Filtered_809, matches("SB\\.C[^K](.*?)_S"))
  WUS_Reg1_No_Gall =  select(Filtered_809, matches("X1(.*?)[^G]_S"))
  WUS_Reg1_Comparisons = list(SB_C, WUS_Reg1_No_Gall)
  # Return list of subsetted data
  return(WUS_Reg1_Comparisons)
}
WUS_Reg1_Comparisons = WUS1_No_Gall_Comparisons_Function(Filtered_809)
SB_C = WUS_Reg1_Comparisons[[1]]
WUS_Reg1_No_Gall = WUS_Reg1_Comparisons[[2]]
```

```
# Specify how to merge
Counts = merge(SB_C, WUS_Reg1_No_Gall, by = 'row.names')
rm(SB_C, WUS_Reg1_No_Gall)

# Make grouping for treatment groups.
Counts = Counts[-(1:2),]
rownames(Counts) <- Counts$Row.names
Counts = subset(Counts, select = -c(Row.names))
my_grouping = c(tail(Counts, 1)) # transformed vs untransformed grouping declared
Counts = head(Counts, -1) # drop Transformed row

Counts = as.matrix.data.frame(Counts)
x = rownames(Counts)
# We need the gene names for later
Gene_Row_Key = data.frame("Num"=rownames(as.data.frame(x)),"Gene"=x)
rm(x)

Counts = apply(Counts, 2, as.numeric)
D = DGEList(Counts, group = unlist(my_grouping))
D = calcNormFactors(D)
D_Samples = D$samples

D = estimateCommonDisp(D)
D = estimateTagwiseDisp(D)
Fish_Exact = exactTest(D)
topTags = topTags(Fish_Exact)

# P = 0.05
simplified_DGE = decideTestsDGE(Fish_Exact, p=0.05, adjust="BH")
my_test = rownames(Fish_Exact)[as.logical(simplified_DGE)]
jpeg('WUS_Reg1_NoGall_vs_SBC_Smear.png')
plotSmear(Fish_Exact, de.tags=my_test, main = "WUS Reg 1 wo Galls vs. Untransformed Cont
abline(h=c(0,10))
dev.off()
```

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

```
# Write summary table
write.table(summary(simplified_DGE), file = 'WUS_Reg1_NoGall_vs_SBC_Summary.txt', quote

# Write direction of differntial expression table
simplified_DGE_frame = data.frame(simplified_DGE)
colnames(simplified_DGE_frame) = 'Direction_Differentially_Regulated'
```

```
row.names(simplified_DGE_frame) = Gene_Row_Key$Gene
WUS_Reg1_NoGall_SBC_Direction = as_tibble(rownames_to_column(simplified_DGE_frame, var =
write_tsv(WUS_Reg1_NoGall_SBC_Direction, 'WUS_Reg1_NoGall_vs_SBC_Direction.tsv')

# Write full output
row.names(Fish_Exact) = Gene_Row_Key$Gene
Fish_tbl = as_tibble(rownames_to_column(Fish_Exact$table, var = 'Gene_Name'))
write_tsv(Fish_tbl, 'WUS_Reg1_NoGall_vs_SBC_Full.tsv')

# Clear workspace
rm(Gene_Row_Key, D, D_Samples, Fish_Exact, my_grouping, Counts, topTags, Fish_tbl, WUS_R
```

WUS2 vs SBC Comparisons

```
Filtered_809 = Filter_809(input_data_809)
# Switch to output folder
setwd('/home/scott/Documents/Uni/Research/Projects/Grape_RNA/Output/Project_2')
#-----------------------------------
WUS2_No_Gall_Comparisons_Function = function(Filtered_809) {
  # Subset the Data
  SB_C = select(Filtered_809, matches("SB\\.C[^K](.*?)_S"))
  WUS_Reg2_No_Gall =  select(Filtered_809, matches("X2(.*?)[^G]_S"))
  WUS_Reg2_Comparisons = list(SB_C, WUS_Reg2_No_Gall)
  # Return list of subsetted data
  return(WUS_Reg2_Comparisons)
}
WUS_Reg2_Comparisons = WUS2_No_Gall_Comparisons_Function(Filtered_809)
SB_C = WUS_Reg2_Comparisons[[1]]
WUS_Reg2_No_Gall = WUS_Reg2_Comparisons[[2]]

# Specify how to merge
Counts = merge(SB_C, WUS_Reg2_No_Gall, by = 'row.names')
rm(SB_C, WUS_Reg2_No_Gall)

# Make grouping for treatment groups.
Counts = Counts[-(1:2),]
rownames(Counts) <- Counts$Row.names
Counts = subset(Counts, select = -c(Row.names))
my_grouping = c(tail(Counts, 1)) # transformed vs untransformed grouping declared
Counts = head(Counts, -1) # drop Transformed row

Counts = as.matrix.data.frame(Counts)
```

```
x = rownames(Counts)
# We need the gene names for later
Gene_Row_Key = data.frame("Num"=rownames(as.data.frame(x)),"Gene"=x)
rm(x)

Counts = apply(Counts, 2, as.numeric)
D = DGEList(Counts, group = unlist(my_grouping))
D = calcNormFactors(D)
D_Samples = D$samples

D = estimateCommonDisp(D)
D = estimateTagwiseDisp(D)
Fish_Exact = exactTest(D)
topTags = topTags(Fish_Exact)

# P = 0.05
simplified_DGE = decideTestsDGE(Fish_Exact, p=0.05, adjust="BH")
my_test = rownames(Fish_Exact)[as.logical(simplified_DGE)]
jpeg('WUS_Reg2_NoGall_vs_SBC_Smear.png')
plotSmear(Fish_Exact, de.tags=my_test, main = "WUS Reg 2 wo Galls vs. Untransformed Cont
abline(h=c(0,10))
dev.off()
```

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

```
# Write summary table
write.table(summary(simplified_DGE), file = 'WUS_Reg2_NoGall__vs_SBC_Summary.txt', quote

# Write direction of differntial expression table
simplified_DGE_frame = data.frame(simplified_DGE)
colnames(simplified_DGE_frame) = 'Direction_Differentially_Regulated'
row.names(simplified_DGE_frame) = Gene_Row_Key$Gene
WUS_Reg1_NoGall_SBC_Direction = as_tibble(rownames_to_column(simplified_DGE_frame, var =
write_tsv(WUS_Reg1_NoGall_SBC_Direction, 'WUS_Reg2_NoGall__vs_SBC_Direction.tsv')

# Write full output
row.names(Fish_Exact) = Gene_Row_Key$Gene
Fish_tbl = as_tibble(rownames_to_column(Fish_Exact$table, var = 'Gene_Name'))
write_tsv(Fish_tbl, 'WUS_Reg2_NoGall_vs_SBC_Full.tsv')

# Clear workspace
rm(Gene_Row_Key, D, D_Samples, Fish_Exact, my_grouping, Counts, topTags, Fish_tbl, WUS_R
```

LFY1 vs SBC Comparisons

```
Filtered_809 = Filter_809(input_data_809)
# Switch to output folder
setwd('/home/scott/Documents/Uni/Research/Projects/Grape_RNA/Output/Project_2')
#-----------------------------------
LFY1_No_Gall_Comparisons_Function = function(Filtered_809) {
  # Subset the Data
  SB_C = select(Filtered_809, matches("SB\\.C[^K](.*?)_S"))
  LFY_Reg1_No_Gall =  select(Filtered_809, matches("X3(.*?)[^G]_S"))
  LFY_Reg1_Comparisons = list(SB_C, LFY_Reg1_No_Gall)
  # Return list of subsetted data
  return(LFY_Reg1_Comparisons)
}
LFY_Reg1_Comparisons = LFY1_No_Gall_Comparisons_Function(Filtered_809)
SB_C = LFY_Reg1_Comparisons[[1]]
LFY_Reg1_No_Gall = LFY_Reg1_Comparisons[[2]]

# Specify how to merge
Counts = merge(SB_C, LFY_Reg1_No_Gall, by = 'row.names')
rm(SB_C, LFY_Reg1_No_Gall)

# Make grouping for treatment groups.
Counts = Counts[-(1:2),]
rownames(Counts) <- Counts$Row.names
Counts = subset(Counts, select = -c(Row.names))
my_grouping = c(tail(Counts, 1)) # transformed vs untransformed grouping declared
Counts = head(Counts, -1) # drop Transformed row

Counts = as.matrix.data.frame(Counts)
x = rownames(Counts)
# We need the gene names for later
Gene_Row_Key = data.frame("Num"=rownames(as.data.frame(x)),"Gene"=x)
rm(x)

Counts = apply(Counts, 2, as.numeric)
D = DGEList(Counts, group = unlist(my_grouping))
D = calcNormFactors(D)
D_Samples = D$samples

D = estimateCommonDisp(D)
D = estimateTagwiseDisp(D)
Fish_Exact = exactTest(D)
topTags = topTags(Fish_Exact)
```

```
# P = 0.05
simplified_DGE = decideTestsDGE(Fish_Exact, p=0.05, adjust="BH")
my_test = rownames(Fish_Exact)[as.logical(simplified_DGE)]
jpeg('LFY_Reg1_NoGall_vs_SBC_Smear.png')
plotSmear(Fish_Exact, de.tags=my_test, main = "LFY Reg 1 wo Galls vs. Untransformed Cont
abline(h=c(0,10))
dev.off()
```

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

```
# Write summary table
write.table(summary(simplified_DGE), file = 'LFY_Reg1_NoGall_vs_SBC_Summary.txt', quote

# Write direction of differntial expression table
simplified_DGE_frame = data.frame(simplified_DGE)
colnames(simplified_DGE_frame) = 'Direction_Differentially_Regulated'
row.names(simplified_DGE_frame) = Gene_Row_Key$Gene
WUS_Reg1_NoGall_SBC_Direction = as_tibble(rownames_to_column(simplified_DGE_frame, var =
write_tsv(WUS_Reg1_NoGall_SBC_Direction, 'LFY_Reg1_NoGall_vs_SBC_Direction.tsv')

# Write full output
row.names(Fish_Exact) = Gene_Row_Key$Gene
Fish_tbl = as_tibble(rownames_to_column(Fish_Exact$table, var = 'Gene_Name'))
write_tsv(Fish_tbl, 'LFY_Reg1_NoGall_vs_SBC_Full.tsv')

# Clear workspace
rm(Gene_Row_Key, D, D_Samples, Fish_Exact, my_grouping, Counts, topTags, Fish_tbl, LFY_R
```

LFY2 vs SBC Comparisons

```
Filtered_809 = Filter_809(input_data_809)
# Switch to output folder
setwd('/home/scott/Documents/Uni/Research/Projects/Grape_RNA/Output/Project_2')
#-----------------------------------
LFY2_No_Gall_Comparisons_Function = function(Filtered_809) {
  # Subset the Data
  SB_C = select(Filtered_809, matches("SB\\.C[^K](.*?)_S"))
  LFY_Reg2_No_Gall =  select(Filtered_809, matches("X4(.*?)[^G]_S"))
  LFY_Reg2_Comparisons = list(SB_C, LFY_Reg2_No_Gall)
  # Return list of subsetted data
  return(LFY_Reg2_Comparisons)
}
```

```
LFY_Reg2_Comparisons = LFY2_No_Gall_Comparisons_Function(Filtered_809)
SB_C = LFY_Reg2_Comparisons[[1]]
LFY_Reg2_No_Gall = LFY_Reg2_Comparisons[[2]]

# Specify how to merge
Counts = merge(SB_C, LFY_Reg2_No_Gall, by = 'row.names')
rm(SB_C, LFY_Reg2_No_Gall)

# Make grouping for treatment groups.
Counts = Counts[-(1:2),]
rownames(Counts) <- Counts$Row.names
Counts = subset(Counts, select = -c(Row.names))
my_grouping = c(tail(Counts, 1)) # transformed vs untransformed grouping declared
Counts = head(Counts, -1) # drop Transformed row

Counts = as.matrix.data.frame(Counts)
x = rownames(Counts)
# We need the gene names for later
Gene_Row_Key = data.frame("Num"=rownames(as.data.frame(x)),"Gene"=x)
rm(x)

Counts = apply(Counts, 2, as.numeric)
D = DGEList(Counts, group = unlist(my_grouping))
D = calcNormFactors(D)
D_Samples = D$samples

D = estimateCommonDisp(D)
D = estimateTagwiseDisp(D)
Fish_Exact = exactTest(D)
topTags = topTags(Fish_Exact)

# P = 0.05
simplified_DGE = decideTestsDGE(Fish_Exact, p=0.05, adjust="BH")
my_test = rownames(Fish_Exact)[as.logical(simplified_DGE)]
jpeg('LFY_Reg2_NoGall_vs_SBC_Smear.png')
plotSmear(Fish_Exact, de.tags=my_test, main = "LFY Reg 2 wo Galls vs. Untransformed Cont
abline(h=c(0,10))
dev.off()
```

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

```
# Write summary table
write.table(summary(simplified_DGE), file = 'LFY_Reg2_NoGall_vs_SBC_Summary.txt', quote
```

```
# Write direction of differntial expression table
simplified_DGE_frame = data.frame(simplified_DGE)
colnames(simplified_DGE_frame) = 'Direction_Differentially_Regulated'
row.names(simplified_DGE_frame) = Gene_Row_Key$Gene
WUS_Reg1_NoGall_SBC_Direction = as_tibble(rownames_to_column(simplified_DGE_frame, var =
write_tsv(WUS_Reg1_NoGall_SBC_Direction, 'LFY_Reg2_NoGall_vs_SBC_Direction.tsv')

# Write full output
row.names(Fish_Exact) = Gene_Row_Key$Gene
Fish_tbl = as_tibble(rownames_to_column(Fish_Exact$table, var = 'Gene_Name'))
write_tsv(Fish_tbl, 'LFY_Reg2_NoGall_vs_SBC_Full.tsv')

# Clear workspace
rm(Gene_Row_Key, D, D_Samples, Fish_Exact, my_grouping, Counts, topTags, Fish_tbl, LFY_R
```

SB-C vs Self — Incomplete Optional Analysis