Example RNA-seq Report: Differences between Breast and Colon Tissue

Example analysis steps/calculations and visualizations for RNA sequencing (RNA-seq) analysis.

Read / Wrangle Files

Read Files in Recursively

```
# get list of files
list.files(path = 'count_tables',
           pattern = 'counts.txt',
           full.names = T) -> files

# Read files in recursively
vroom(file = files, id = 'file_path', delim = '\t', comment = '#', skip = 2,
      col_names = c('ensembl_gene_id', 'chr', 'start', 'end',
                    'strand', 'length', 'count')) %>%
  mutate(sample_id = str_extract(file_path, 'ENCFF[0-9,A-Z]{6}'),
         tissue = ifelse(str_detect(file_path, 'breast'),
                         'breast', 'colon')) %>%
  dplyr::select(-file_path) -> data
```

```
## Rows: 406,357
## Columns: 8
## Delimiter: "\t"
## chr [5]: ensembl_gene_id, chr, start, end, strand
## dbl [2]: length, count
##
## Use `spec()` to retrieve the guessed column specification
## Pass a specification to the `col_types` argument to quiet this message
```

Wrangle Data for DESeq2

Transform the data into a count matrix and an annotation table.

```
data %>%
  dplyr::select(ensembl_gene_id, sample_id, count) %>%
  pivot_wider(names_from = sample_id, values_from = count) %>%
  as.data.frame() -> count_matrix

data %>%
  dplyr::select(sample_id, tissue) %>%
  distinct() -> metadata
```

Convert to DESeq2 DESeqDataSet Object

```
dds <- DESeqDataSetFromMatrix(countData = count_matrix,
                              colData = metadata,
```

```
                                tidy = TRUE,
                                design = ~ tissue)
```

```
## converting counts to integer mode
```

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

Pre-filter the dataset

Remove rows that contain only zero counts. There's no point in testing genes where nothing was detected.

```
### filter out rows that contain only zero counts
keep <- rowSums(counts(dds)) > 1
dds <- dds[keep, ]
```

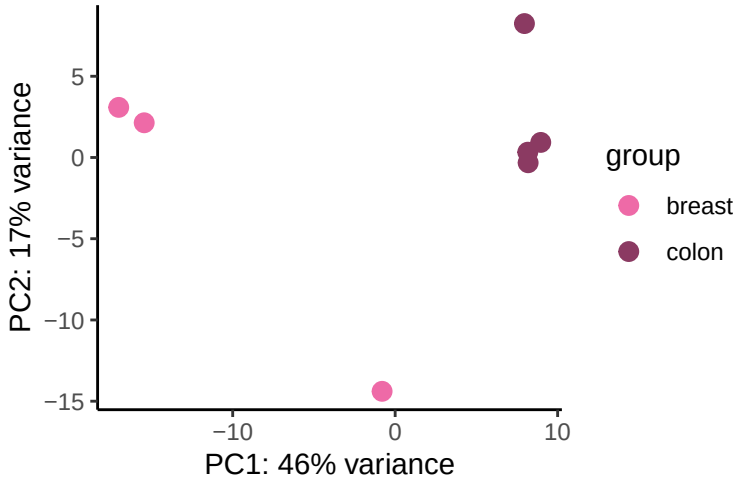
Check Quality by Examing the Associations Between Samples

PCA

Check to make sure that our groups are distinct from one another and there's no obvious clustering by technical effects. As seen in the PCA plot generated below, PC1 and PC2 separate samples by tissue type, the effect we expect to see.

```
# normalize the counts
vsd <- varianceStabilizingTransformation(dds, blind = FALSE)

# plot the PCa
plotPCA(vsd, intgroup = 'tissue') +
  geom_point() +
  scale_color_manual(values = c('hotpink2', 'hotpink4')) +
  theme_classic()
```



Differential Expression

Calculate differential expression

```
dds <- DESeq(dds)
```

```
## estimating size factors
## estimating dispersions
## gene-wise dispersion estimates
## mean-dispersion relationship
## final dispersion estimates
## fitting model and testing
```

Wrangle and Save the Results

Get human readable gene names

```
# check which annotations are available.
columns(org.Hs.eg.db)
```

```
##  [1] "ACCNUM"       "ALIAS"        "ENSEMBL"      "ENSEMBLPROT"  "ENSEMBLTRANS"
##  [6] "ENTREZID"     "ENZYME"       "EVIDENCE"     "EVIDENCEALL"  "GENENAME"
## [11] "GO"           "GOALL"        "IPI"          "MAP"          "OMIM"
## [16] "ONTOLOGY"     "ONTOLOGYALL"  "PATH"         "PFAM"         "PMID"
## [21] "PROSITE"      "REFSEQ"       "SYMBOL"       "UCSCKG"       "UNIGENE"
## [26] "UNIPROT"
```

```
# use the Ensembl IDs to find the corresponding HGNC IDs
mapIds(org.Hs.eg.db,
       keys = rownames(results(dds)),
       column = 'SYMBOL',
       keytype = 'ENSEMBL',
       multiVals = 'first') %>%
  enframe(name = 'ensembl_gene_id', value = 'gene') -> gene_names
```

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

Wrangle the differential expression result object to get a rectangular table with the human readable gene IDs.

```
results(dds) %>%
  as.data.frame() %>%
  rownames_to_column('ensembl_gene_id') %>%
  left_join(gene_names, by = 'ensembl_gene_id') %>%
  dplyr::select(gene, ensembl_gene_id, everything()) %>%
  mutate(sig = ifelse(padj < 0.05 & abs(log2FoldChange) >= 1,
                      'sig', 'notsig'),
         log_qvalue = -log10(padj)) %>%
  replace_na(list(sig = 'notsig', log_qvalue = 0)) -> diff_exp_tbl
```

Save the results of the differential expression test.

```
# write_tsv(diff_exp_tbl, 'diff_exp.tsv')
```

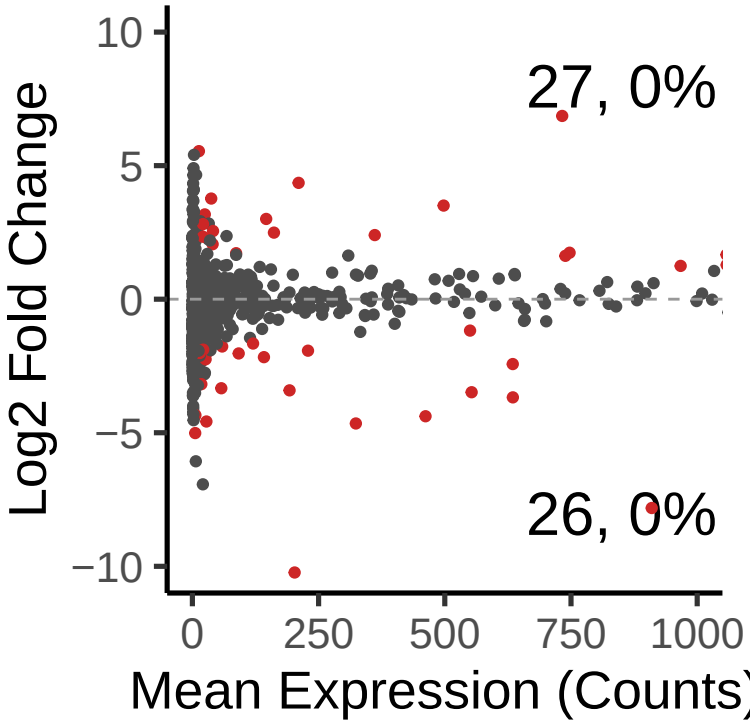
Visualize Results

MA Plot

Only a few genes are differentially expressed, many of them with originally low counts.

```
diff_exp_tbl %>%
  mutate(direction = ifelse(log2FoldChange < 0, 'down', 'up')) %>%
  group_by(direction, sig) %>%
  dplyr::count() %>%
  ungroup() %>%
  filter(sig == 'sig') %>%
  mutate(label = paste0(n, ', ', round((n / nrow(diff_exp_tbl)), 1), '%'),
         baseMean = 850,
         log2FoldChange = c(-8, 8)) -> ma_labels

# plot
ggplot(diff_exp_tbl, aes(x = baseMean, y = log2FoldChange)) +
  geom_text(data = ma_labels, aes(label = label), size = 8) +
  geom_point(aes(color = sig)) +
  scale_color_manual(values = c('gray30', 'firebrick3')) +
  geom_hline(yintercept = 0, color = 'gray60', linetype = 'dashed') +
  labs(x = 'Mean Expression (Counts)', y = 'Log2 Fold Change') +
  coord_cartesian(xlim = c(0, 1000), ylim = c(-10, 10)) +
  theme_classic(base_size = 20) +
  theme(legend.position = 'none')
```

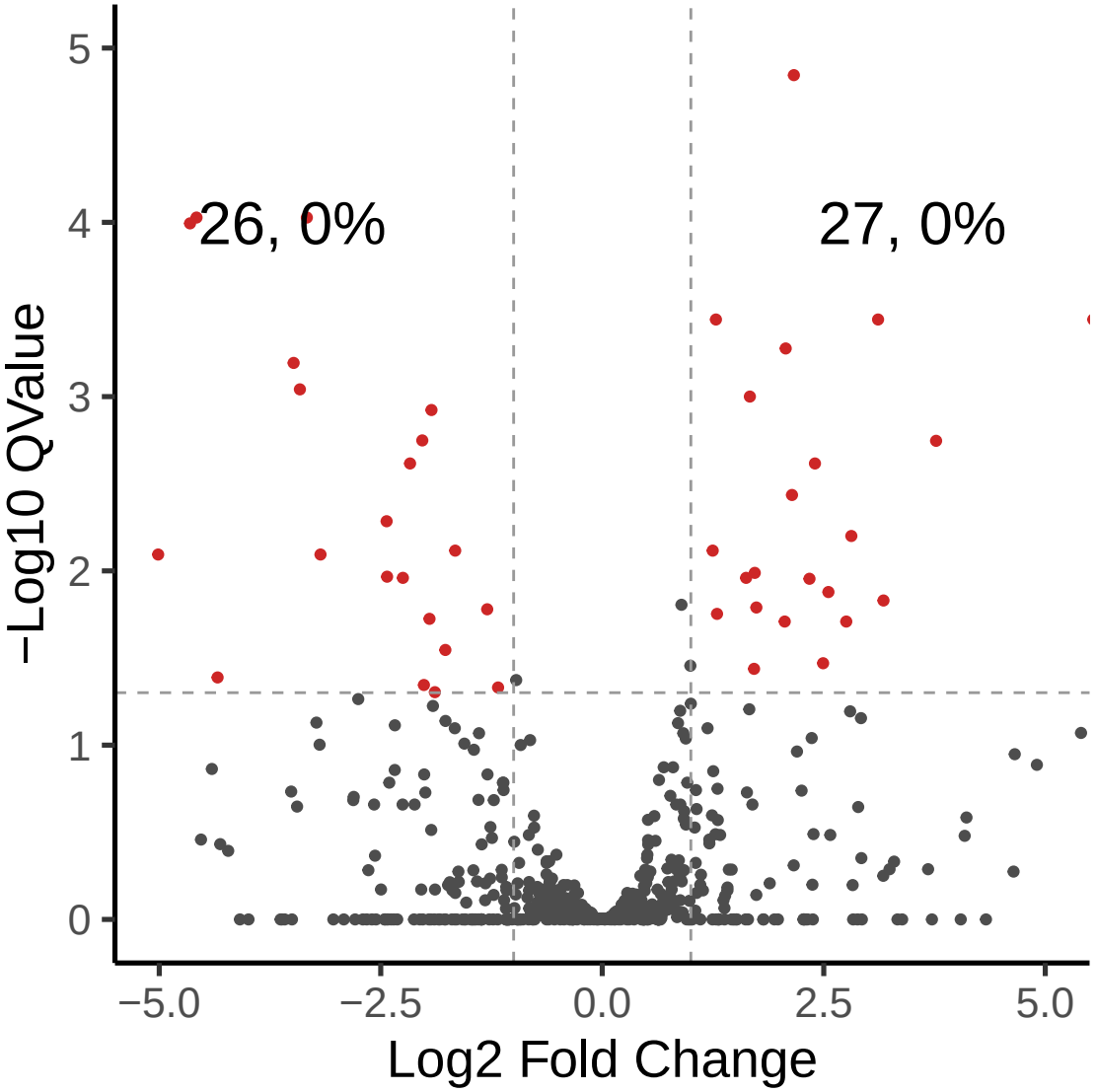


Volcano Plot

Only a few up- and down-regulated genes.

```r
# Create labels for the number and percentage of significantly up- and down-
# regulated genes
diff_exp_tbl %>%
  mutate(direction = ifelse(log2FoldChange < 0, 'down', 'up')) %>%
  group_by(direction, sig) %>%
  dplyr::count() %>%
  ungroup() %>%
  # complete(direction, sig, fill = list(n = 0)) %>%
  # na.omit() %>%
  filter(sig == 'sig') %>%
  mutate(label = paste0(n, ', ', round((n / nrow(diff_exp_tbl)), 1), '%'),
         log2FoldChange = c(-3.5, 3.5),
         log_qvalue = 4) -> volc_labels

# plot
ggplot(diff_exp_tbl, aes(x = log2FoldChange, y = log_qvalue)) +
  geom_point(aes(color = sig)) +
  scale_color_manual(values = c('gray30', 'firebrick3')) +
  geom_hline(yintercept = -log10(0.05), color = 'gray60', linetype = 'dashed') +
  geom_vline(xintercept = c(-1, 1), color = 'gray60', linetype = 'dashed') +
  geom_text(data = volc_labels, aes(label = label), size = 8) +
  labs(x = 'Log2 Fold Change', y = '-Log10 QValue') +
  coord_cartesian(xlim = c(-5, 5), ylim = c(0, 5)) +
  theme_classic(base_size = 20) +
  theme(legend.position = 'none')
```



Pathway Analysis

Wrangle Differential Expression Data for Pathway Analysis

Get the list of Entrez IDs that correspond to all our genes detected in the differential expression analysis.

```
# get Entrez IDs from our Ensembl IDs for fgsea()
AnnotationDbi::mapIds(org.Hs.eg.db,
                      keys = unique(diff_exp_tbl$ensembl_gene_id),
                      keytype = 'ENSEMBL',
                      column = 'ENTREZID',
                      multiVals = 'first') %>%
  enframe(name = 'gene', value = 'entrez_id') %>%
  unnest(c(entrez_id)) -> entrez_ids
```

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

```
# make a named vector of our genes for fgsea()
diff_exp_tbl %>%
  dplyr::select(ensembl_gene_id, stat) %>%
  left_join(entrez_ids, by = c('ensembl_gene_id' = 'gene')) %>%
  dplyr::select(entrez_id, stat) %>%
  deframe() -> diff_exp_res
```

Calculate Pathway Analysis

Load the Reactome pathways

```
# use Reactome pathwayse
reactome_pathways <- reactomePathways(names(exampleRanks))
```

```
## Loading required namespace: reactome.db
```

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

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

Run the pathway analysis

```
fgsea_res <- fgsea(pathways = reactome_pathways,
                   stats = exampleRanks,
                   nperm = 1000)
```

Save results

```
### save results
fgsea_res %>%
  unnest(c(leadingEdge)) #%>%
```

```
## # A tibble: 17,134 x 8
##    pathway               pval  padj     ES    NES nMoreExtreme  size leadingEdge
##    <chr>                <dbl> <dbl>  <dbl>  <dbl>        <dbl> <int> <chr>
##  1 5-Phosphoribose 1-d~ 0.889 0.954  0.427  0.666          454     2 328099
##  2 5-Phosphoribose 1-d~ 0.889 0.954  0.427  0.666          454     2 19139
##  3 A tetrasaccharide l~ 0.657 0.856  0.322  0.837          363    11 14733
##  4 A tetrasaccharide l~ 0.657 0.856  0.322  0.837          363    11 20971
##  5 A tetrasaccharide l~ 0.657 0.856  0.322  0.837          363    11 20970
##  6 A tetrasaccharide l~ 0.657 0.856  0.322  0.837          363    11 12032
##  7 A tetrasaccharide l~ 0.657 0.856  0.322  0.837          363    11 29873
##  8 A tetrasaccharide l~ 0.657 0.856  0.322  0.837          363    11 218271
##  9 A tetrasaccharide l~ 0.657 0.856  0.322  0.837          363    11 13179
## 10 Abacavir metabolism  0.285 0.640 -0.628 -1.16           135     3 11522
## # ... with 17,124 more rows
```

```
  # write_tsv('all_pathway_results.tsv')

### save collapsed results
# find the essential top-level pathways
collapsed_pathways <- collapsePathways(fgsea_res,
                                       pathways = reactome_pathways,
                                       stats = exampleRanks)
```

```
## Warning in fgsea(pathways = pathways[pathwaysToCheck], stats = stats[u2], :
```

```
## There were 2 pathways for which P-values were not calculated properly due to
## unbalanced gene-level statistic values

## Warning in fgsea(pathways = pathways[pathwaysToCheck], stats = stats[u2], :
## There were 2 pathways for which P-values were not calculated properly due to
## unbalanced gene-level statistic values

## Warning in fgsea(pathways = pathways[pathwaysToCheck], stats = stats[u2], :
## There were 3 pathways for which P-values were not calculated properly due to
## unbalanced gene-level statistic values

## Warning in fgsea(pathways = pathways[pathwaysToCheck], stats = stats[u2], :
## There were 1 pathways for which P-values were not calculated properly due to
## unbalanced gene-level statistic values
```

```
# filter the results for the essential pathways
fgsea_res %>%
  filter(pathway %in% collapsed_pathways$mainPathways) %>%
  arrange(pathway) %>%
  unnest(c(leadingEdge)) #%>%
```

```
## # A tibble: 1,573 x 8
##    pathway               pval   padj     ES   NES nMoreExtreme  size leadingEdge
##    <chr>                <dbl>  <dbl>  <dbl> <dbl>        <dbl> <int> <chr>
##  1 Activated NOTCH1 ~ 0.0112  0.0975 -0.763 -1.84            4     7 14357
##  2 Activated NOTCH1 ~ 0.0112  0.0975 -0.763 -1.84            4     7 18128
##  3 Activated NOTCH1 ~ 0.0112  0.0975 -0.763 -1.84            4     7 74198
##  4 Apoptosis          0.00154 0.0278  0.516  2.09            0    71 58801
##  5 Apoptosis          0.00154 0.0278  0.516  2.09            0    71 14958
##  6 Apoptosis          0.00154 0.0278  0.516  2.09            0    71 97165
##  7 Apoptosis          0.00154 0.0278  0.516  2.09            0    71 22352
##  8 Apoptosis          0.00154 0.0278  0.516  2.09            0    71 12043
##  9 Apoptosis          0.00154 0.0278  0.516  2.09            0    71 14103
## 10 Apoptosis          0.00154 0.0278  0.516  2.09            0    71 269582
## # ... with 1,563 more rows
```

```
  # write_tsv('top_level_pathway_results.tsv')
```

Visualize

Look at the top 10 most up- and down-regulated statistically significant pathways.

```
fgsea_res %>% #arrange(desc(nchar(pathway)))
# filter for significant pathways
  filter(padj < 0.05) %>%
# arrange by the normalized enrichment score
  arrange(NES) %>%
# get the first and last 10 rows which will be the 10 most up- and down-
# regulated pathways
  do(rbind(head(., 10), tail(., 10))) %>%
# the pathway names can be long, so if they're over 20 characters, subset them
# otherwise use the whole name
  mutate(pathway_short = ifelse(nchar(pathway) <= 30,
                                pathway,
                                paste0(str_sub(pathway, start = 1, end = 27),
                                       '...'))) %>%
```

```
ggplot(aes(x = reorder(pathway_short, NES), y = NES)) +
  geom_col(aes(fill = padj)) +
  scale_fill_viridis(direction = -1) +
  coord_flip() +
  labs(x = 'Pathway',
       y = 'Normalized Enrichment Score (NES)',
       fill = 'Q-Value') +
  theme_minimal(base_size = 16)
```