MEDICAL MICROBIOLOGY AND INFECTIOUS DISEASES CODING WORKSHOP

Presents

RNAseq data analysis

INSTRUCTED BY

Jessy Slota

Educational series supported by the Department of MMID to promote knowledge translation and dissemination among students and researchers

INFORMATION FOR PARTICIPANTS

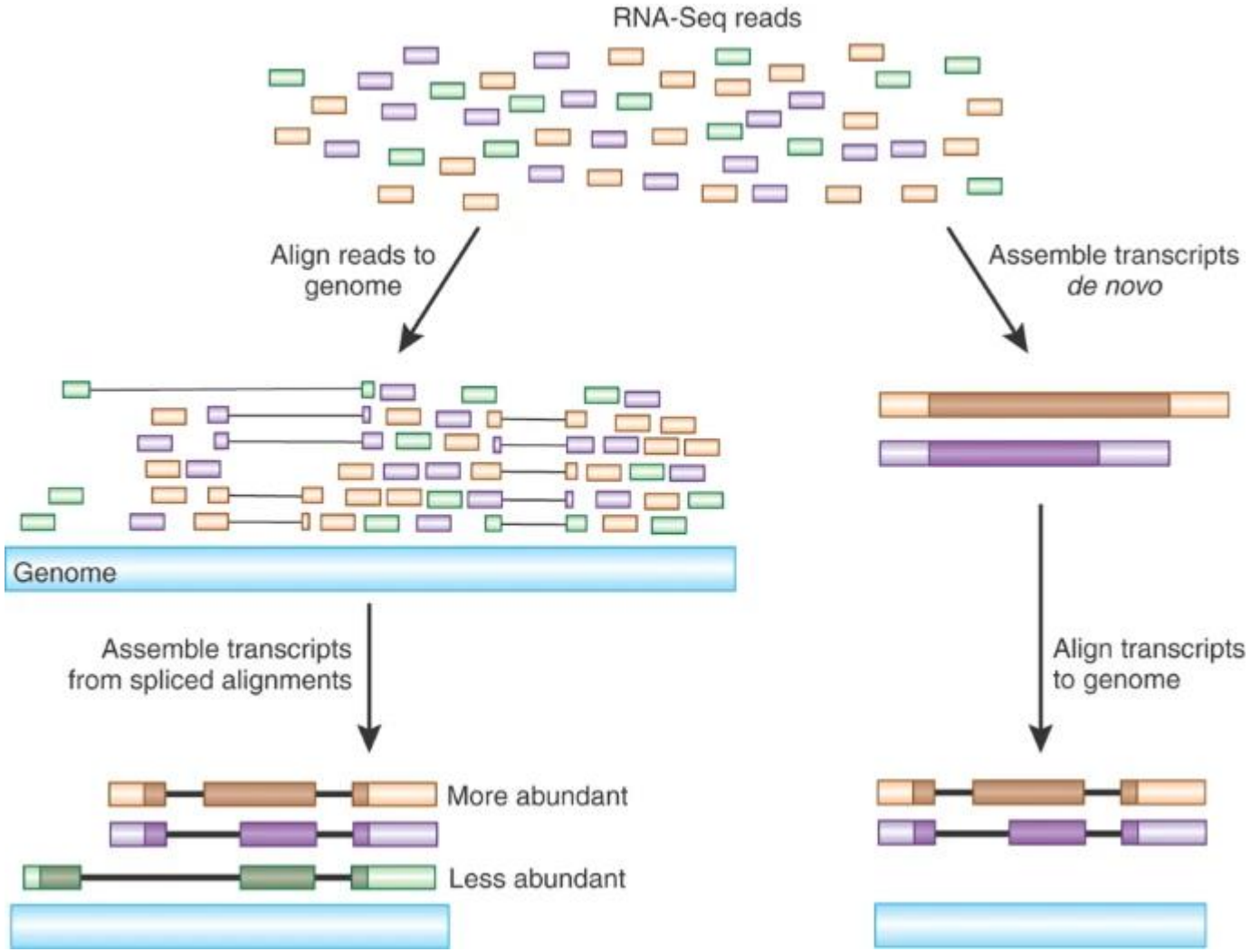
All workshops are being recorded and posted to the MMID Coding Workshop - YouTube

Question and Answer period will not be recorded.

LEARNING OBJECTIVES

1. *Raw read count processing (Script 1)*
2. *Normalization with DESeq2 (Script 2)*
3. *Differential expression analysis with DESeq2 (Script 3)*
4. *Functional enrichment analysis with Enrichr (Script 4)*
5. *Common data visualizations (Script 5)*

What is RNAseq and why use it?

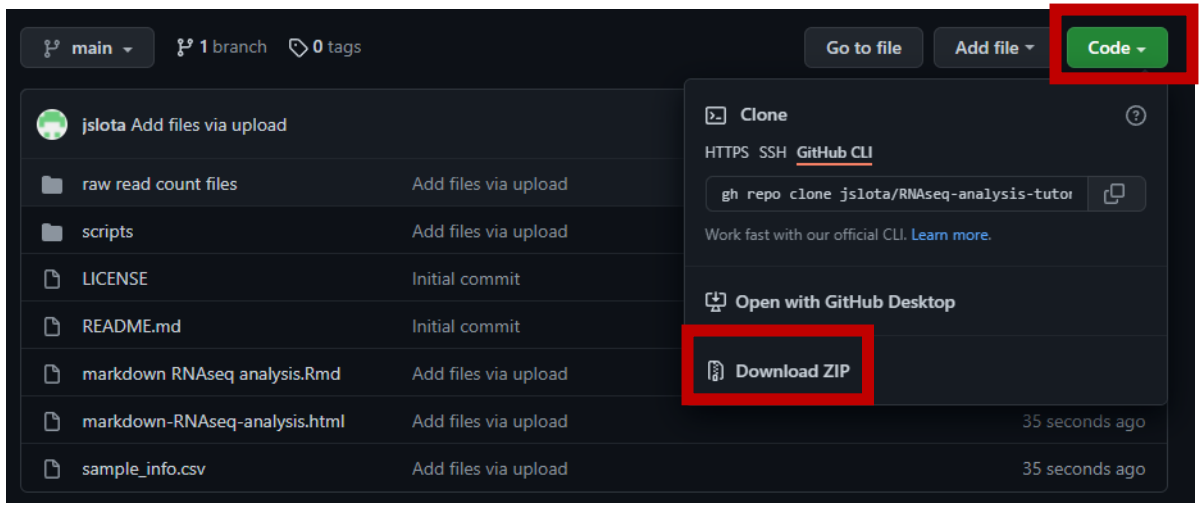


Haas, B., Zody, M. Advancing RNA-Seq analysis. *Nat Biotechnol* **28,** 421–423 (2010). https://doi-org.uml.idm.oclc.org/10.1038/nbt0510-421

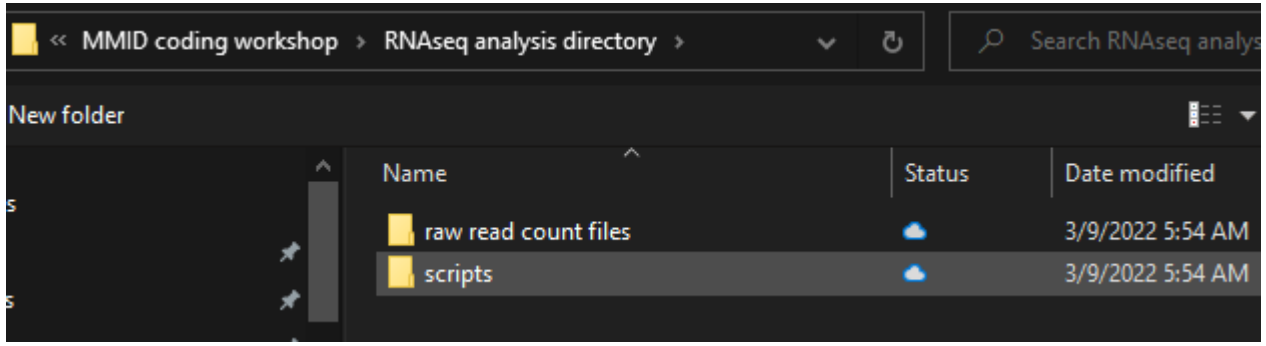
Principles of RNAseq analysis

1. **Pre-processing sequencing fastq data (not covered)**
   - *Taking raw fastq files from sequencing run and processing into read counts for each transcript*
2. **Normalization**
   - *Adjusting read count values for proper statistical analysis and data visualizations*
3. **Differential expression analysis**
   - *Identify transcripts with altered abundance*
4. **Functional enrichment**
   - *Identify pathways or groups of related genes that are enriched with altered transcripts*

Tutorial: Setting up the analysis directory



Download/unzip from github
https://github.com/MMID-coding-workshop/2022-03-09-RNA-seq-data-analysis-in-R



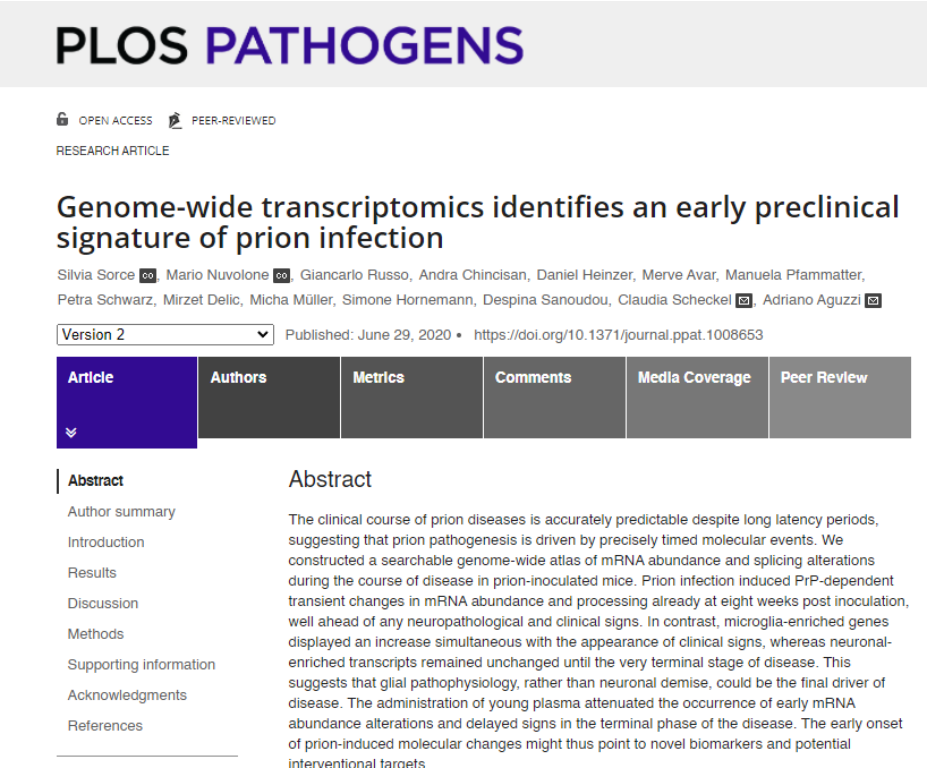
Don't forget to set working directory to "analysis directory"!

Tutorial: Setting up the analysis directory

```
R version 4.1.2 (2021-11-01) -- "Bird Hippie"
Copyright (C) 2021 The R Foundation for Statistical Computing
Platform: x86_64-w64-mingw32/x64 (64-bit)

R is free software and comes with ABSOLUTELY NO WARRANTY.
You are welcome to redistribute it under certain conditions.
Type 'license()' or 'licence()' for distribution details.

R is a collaborative project with many contributors.
Type 'contributors()' for more information and
'citation()' on how to cite R or R packages in publications.

Type 'demo()' for some demos, 'help()' for on-line help, or
'help.start()' for an HTML browser interface to help.
Type 'q()' to quit R.

> setwd("C:/Users/jslota/OneDrive - University of Manitoba/Misc/Bioinformatics lessons/MMID coding workshop/RNAseq analysis director
y")
>
```

Don’t forget to set working directory to “analysis directory”!

The tutorial dataset



PLOS PATHOGENS

OPEN ACCESS PEER-REVIEWED

RESEARCH ARTICLE

Genome-wide transcriptomics identifies an early preclinical signature of prion infection

Silvia Sorce, Mario Nuvolone, Giancarlo Russo, Andra Chincisan, Daniel Heinzer, Merve Avar, Manuela Pfammatter, Petra Schwarz, Mirzet Delic, Micha Müller, Simone Hornemann, Despina Sanoudou, Claudia Scheckel, Adriano Aguzzi

Version 2 Published: June 29, 2020 • https://doi.org/10.1371/journal.ppat.1008653

Article | Authors | Metrics | Comments | Media Coverage | Peer Review

Abstract
Author summary
Introduction
Results
Discussion
Methods
Supporting information
Acknowledgments
References

Abstract

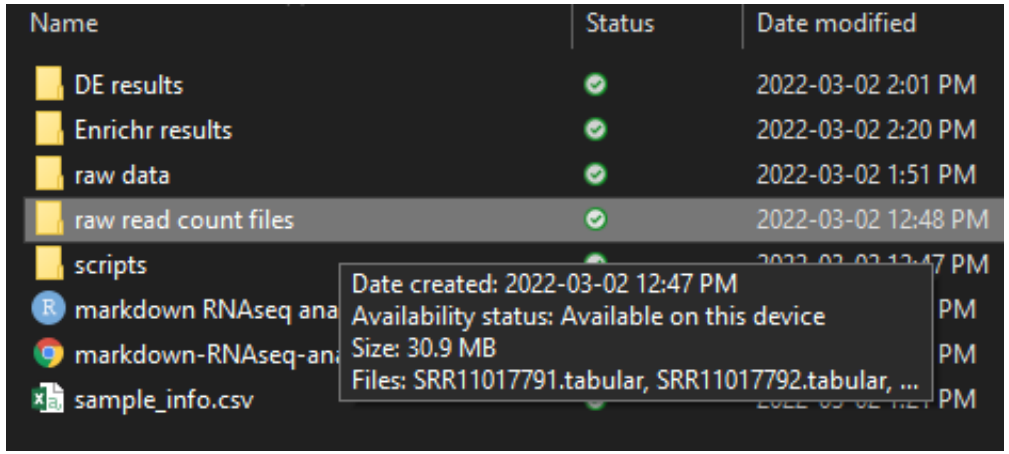
The clinical course of prion diseases is accurately predictable despite long latency periods, suggesting that prion pathogenesis is driven by precisely timed molecular events. We constructed a searchable genome-wide atlas of mRNA abundance and splicing alterations during the course of disease in prion-inoculated mice. Prion infection induced PrP-dependent transient changes in mRNA abundance and processing already at eight weeks post inoculation, well ahead of any neuropathological and clinical signs. In contrast, microglia-enriched genes displayed an increase simultaneous with the appearance of clinical signs, whereas neuronal-enriched transcripts remained unchanged until the very terminal stage of disease. This suggests that glial pathophysiology, rather than neuronal demise, could be the final driver of disease. The administration of young plasma attenuated the occurrence of early mRNA abundance alterations and delayed signs in the terminal phase of the disease. The early onset of prion-induced molecular changes might thus point to novel biomarkers and potential interventional targets

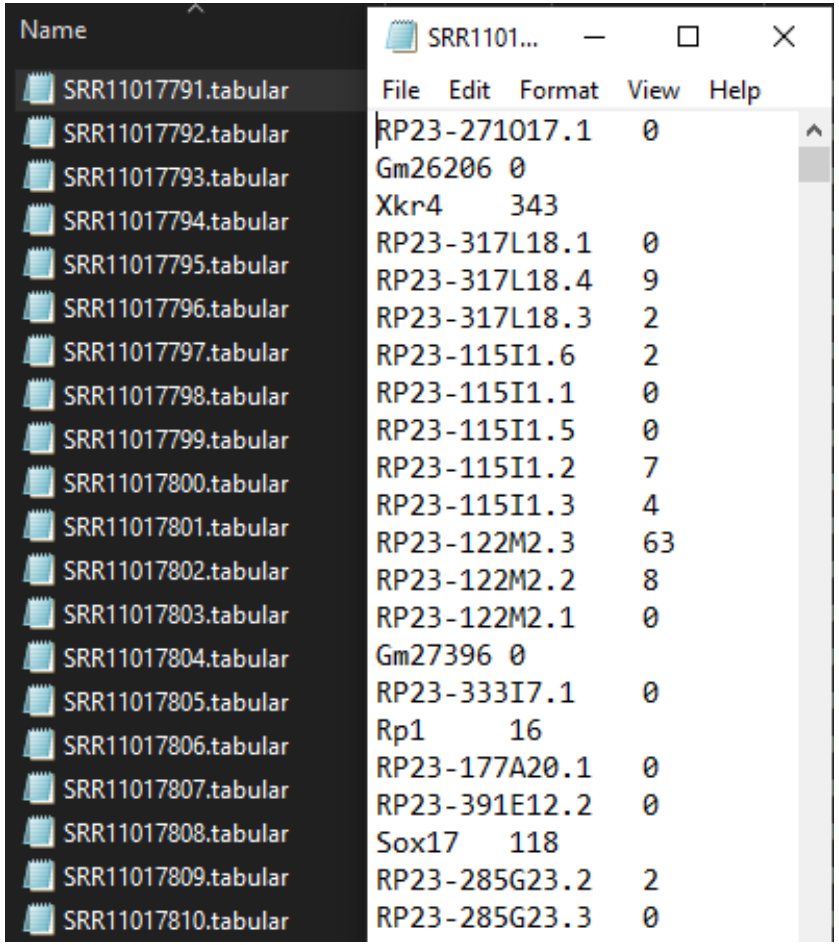
RNAseq data on brain tissue (hippocampus) from mice infected with prions

Timecourse study… evaluating gene expression at multiple timepoints

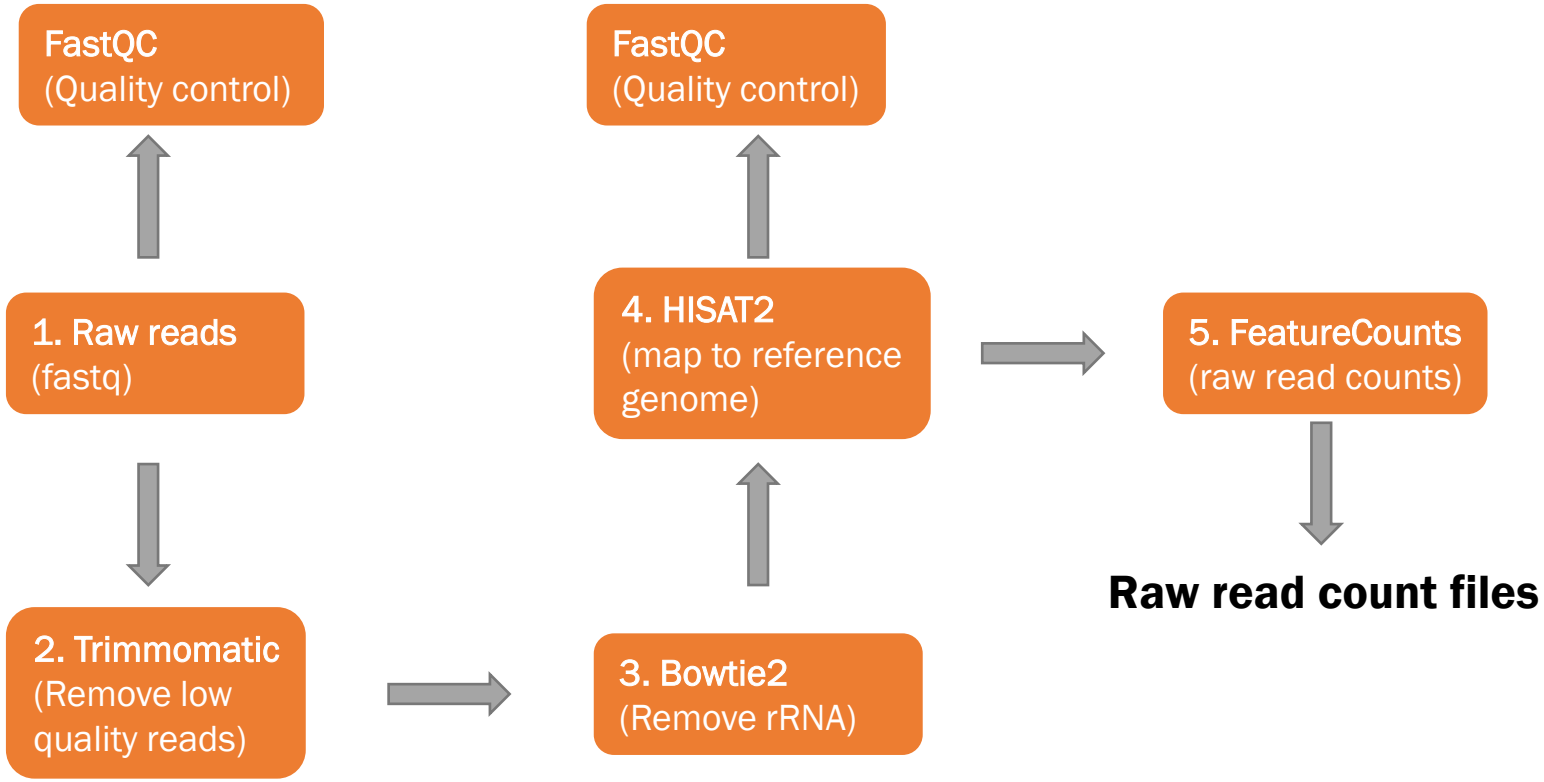
https://journals.plos.org/plospathogens/article?id=10.1371/journal.ppat.1008653

Raw read count files





Pre-processing pipeline *(in Galaxy… not covered here)*



Script 1: Raw Read count processing

```
10
11  ###Collect all raw read count files and merge into one matrix
12  data_files <- Sys.glob("raw read count files/*.tabular") #store paths for all ra
13  tmp <- list() #create an empty list to store each file
14  for (i in data_files) { #for loop to load each individual read count file
15    X <- gsub(".tabular.*", "", gsub(".*raw read count files/", "", i)) #extract s
16    tmp[[X]] <- read.delim(i, row.names = 1, header = FALSE) #load read count file
17    colnames(tmp[[X]]) <- X #rename column with sample name
18    print(X) #print sample name to track progress in console
19  }
20  read_counts <- do.call(cbind, tmp) #do.call function collapses all objects withi
21
22  #Clean up read count file
23  read_counts <- read_counts[rowMeans(read_counts)>0,] # remove all transcripts th
24  read_counts <- read_counts[order(rowMeans(read_counts), decreasing = TRUE),] # o
25
26  #Save files for further analysis
27  if (dir.exists("raw data") == FALSE) { dir.create("raw data") }
28  write.csv(read_counts, "raw data/raw_read_counts.csv")
```

Script 1: Raw Read count processing

	SRR11017791	SRR11017792	SRR11017793	SRR11017794	SRR11017795	SRR11017796	SRR11017797	SRR11017798
mt-Co1	638684	664499	976518	919394	924200	683345	845559	715354
mt-Cytb	185981	274826	363407	281813	364756	235279	311545	276096
Camk2a	171063	178796	311863	219557	264300	164263	388728	229171
mt-Nd1	166391	190562	258089	244864	262878	190919	248941	212294
Slc1a2	126540	126114	182746	157137	153271	130549	152264	131832
mt-Nd5	54220	138239	177209	89367	170449	84789	154445	172270
mt-Nd2	68586	139209	179154	112238	197984	116208	167213	159174
mt-Rnr2	86533	142229	155507	93848	197622	121925	140359	142927
Atp1a3	73864	77605	158591	103131	125274	73056	197820	151542
Kif5a	119507	101976	148837	143190	124609	139305	140241	143170
mt-Nd4	78632	131552	174437	117015	179092	108953	143368	130490
Cpe	98846	90106	165564	106653	131403	101327	134521	112817
Ncdn	54591	66568	128647	90430	102257	53974	197582	112443
Snhg11	74382	86111	128310	84504	136110	95268	119266	126592
Calm1	96670	92803	146995	116109	123161	92017	132319	102127

Script 2: Normalization with DESeq2

Raw read count matrix

	SRR11017791	SRR11017792	SRR11017793	SRR11017794	SRR11017795	SRR11017796	SRR11017797	SRR11017798
mt-Co1	638684	664499	976518	919394	924200	683345	845559	715354
mt-Cytb	185981	274826	363407	281813	364756	235279	311545	276096
Camk2a	171063	178796	311863	219557	264300	164263	388728	229171
mt-Nd1	166391	190562	258089	244864	262878	190919	248941	212294
Slc1a2	126540	126114	182746	157137	153271	130549	152264	131832
mt-Nd5	54220	138239	177209	89367	170449	84789	154445	172270
mt-Nd2	68586	139209	179154	112238	197984	116208	167213	159174
mt-Rnr2	86533	142229	155507	93848	197622	121925	140359	142927
Atp1a3	73864	77605	158591	103131	125274	73056	197820	151542
Kif5a	119507	101976	148837	143190	124609	139305	140241	143170
mt-Nd4	78632	131552	174437	117015	179092	108953	143368	130490
Cpe	98846	90106	165564	106653	131403	101327	134521	112817
Ncdn	54591	66568	128647	90430	102257	53974	197582	112443
Snhg11	74382	86111	128310	84504	136110	95268	119266	126592
Calm1	96670	92803	146995	116109	123161	92017	132319	102127

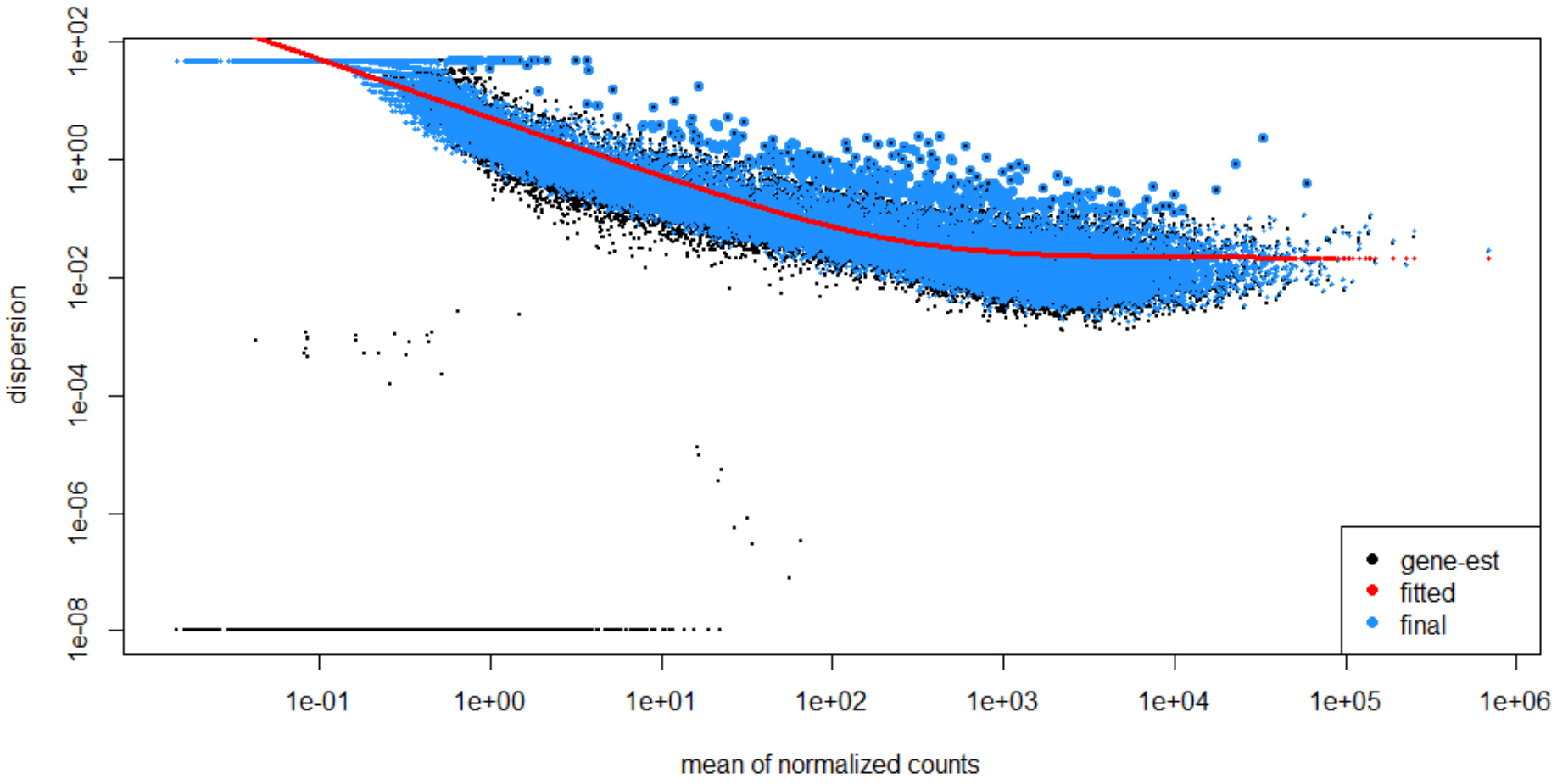
Sample info matrix

	timepoint	treatment
SRR11017791	4_wpi	Mock
SRR11017792	4_wpi	Mock
SRR11017793	4_wpi	Mock
SRR11017794	4_wpi	RML
SRR11017795	4_wpi	RML
SRR11017796	4_wpi	RML
SRR11017797	8_wpi	Mock
SRR11017798	8_wpi	Mock
SRR11017799	8_wpi	Mock
SRR11017800	8_wpi	RML
SRR11017801	8_wpi	RML
SRR11017802	8_wpi	RML
SRR11017803	12_wpi	Mock
SRR11017804	12_wpi	Mock
SRR11017805	12_wpi	Mock
SRR11017806	12_wpi	RML
SRR11017807	12_wpi	RML
SRR11017808	12_wpi	RML

Script 2: Normalization with DESeq2

```
library(DESeq2)
library(ggplot2)
library(RColorBrewer)

#load raw data
read_counts <- read.csv("raw data/raw_read_counts.csv", row.names = 1)#load read count files
sample_info <- read.csv("sample_info.csv", row.names = 2, stringsAsFactors = TRUE)[,-1]#load sample info and

summary(colnames(read_counts)==rownames(sample_info))#make sure samples are in order

#make DEseq data object
dds <- DESeqDataSetFromMatrix(countData = read_counts, colData = sample_info, design = ~treatment+timepoint)
dds <- DESeq(dds)

#plot dispersion estimates to examine normalization
plotDispEsts(dds)
```

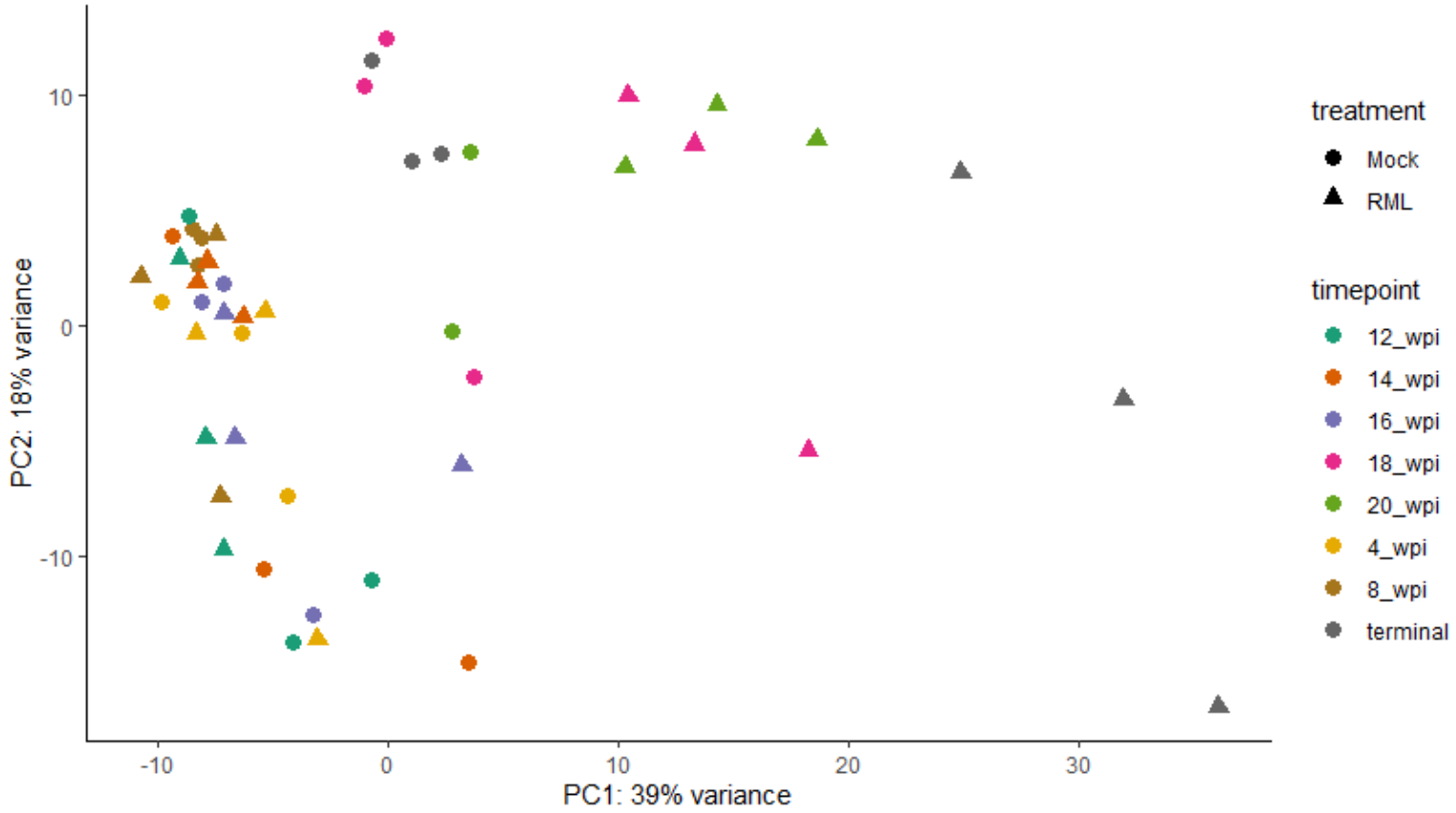
Script 2: Normalization with DESeq2



Script 2: Normalization with DESeq2

```
#Get normalized counts and make PCA plots
norm_counts <- vst(dds)#extract normalized read counts
plotPCA(norm_counts, intgroup=c("treatment", "timepoint"))#make a basic PCA plot

#make a custom PCA plot with ggplot
pcaData <- plotPCA(norm_counts, intgroup=c("treatment", "timepoint"), returnData=TRUE)
percentVar <- round(100 * attr(pcaData, "percentVar"))
ggplot(pcaData, aes(PC1, PC2, color=timepoint, shape=treatment)) +
  geom_point(size=3) +
  xlab(paste0("PC1: ",percentVar[1],"% variance")) +
  ylab(paste0("PC2: ",percentVar[2],"% variance")) +
  scale_color_manual(values = brewer.pal(8, "Dark2")) +
  coord_fixed() +
  theme_classic()

#Save normalized read counts for visualization later
write.csv(assay(norm_counts), "raw data/normalized_read_counts.csv")
```

Script 2: Normalization with DESeq2



Script 3: Differential expression analysis with DESeq2

```
#load raw data
read_counts <- read.csv("raw data/raw_read_counts.csv", row.names = 1)#load read count files
sample_info <- read.csv("sample_info.csv", row.names = 2, stringsAsFactors = TRUE)[,-1]#load sample info and set rownam

#Only keep samples from terminal timepoint
samples <- rownames(sample_info[sample_info$timepoint=="terminal",])

#make DEseq data object
dds <- DESeqDataSetFromMatrix(countData = read_counts[,samples], colData = sample_info[samples,], design = ~treatment)
dds <- DESeq(dds)

#get differential expression results
resultsNames(dds)
res <- results(object = dds, contrast = c("treatment", "RML", "Mock"))#Contrast = RML vs Mock samples

#clean up results file
res <- res[order(res$padj),]
res <- na.omit(res)
res <-as.data.frame(res)

summary(res$padj < 0.05)#Get summary of statistical significance

#Save differential expression results
if (dir.exists("DE results")==FALSE) { dir.create("DE results") }
write.csv(res, "DE results/RML_terminal_DE_results.csv")
```

Script 3: Differential expression analysis with DESeq2

	baseMean	log2FoldChange	lfcSE	stat	pvalue	padj
Gfap	200962.23942	4.099226	0.1888948	21.701103	2.003008e-104	4.306066e-100
A2m	5075.56640	5.196894	0.2398370	21.668442	4.073058e-104	4.378130e-100
Aspg	1164.66046	3.983006	0.1998751	19.927471	2.351501e-88	1.685086e-84
Serpina3n	29093.05234	4.210916	0.2163325	19.465017	2.174211e-84	1.168530e-80
Cybrd1	1074.94323	2.961314	0.1606106	18.437851	6.528469e-76	2.806981e-72
Fcgr2b	1682.84134	4.089544	0.2227112	18.362541	2.620517e-75	9.389312e-72
Lag3	3152.78748	4.092816	0.2322295	17.624013	1.611569e-69	4.949359e-66
Endou	393.06778	3.691634	0.2147660	17.189096	3.204899e-66	8.612364e-63
Serpinf2	702.87876	5.279462	0.3079804	17.142200	7.187929e-66	1.716957e-62
Cxcl10	300.30221	5.717513	0.3371839	16.956657	1.718462e-64	3.694350e-61
Osmr	2540.90090	3.622764	0.2258078	16.043570	6.340280e-58	1.239121e-54
Socs3	429.11333	3.480601	0.2242750	15.519342	2.566694e-54	4.598232e-51
Tlr2	792.08125	4.667687	0.3018894	15.461581	6.303187e-54	1.042353e-50
S1pr3	3022.45898	3.550258	0.2299564	15.438827	8.971651e-54	1.377661e-50
Slc43a3	901.33985	4.933819	0.3220339	15.320805	5.552526e-53	7.957880e-50

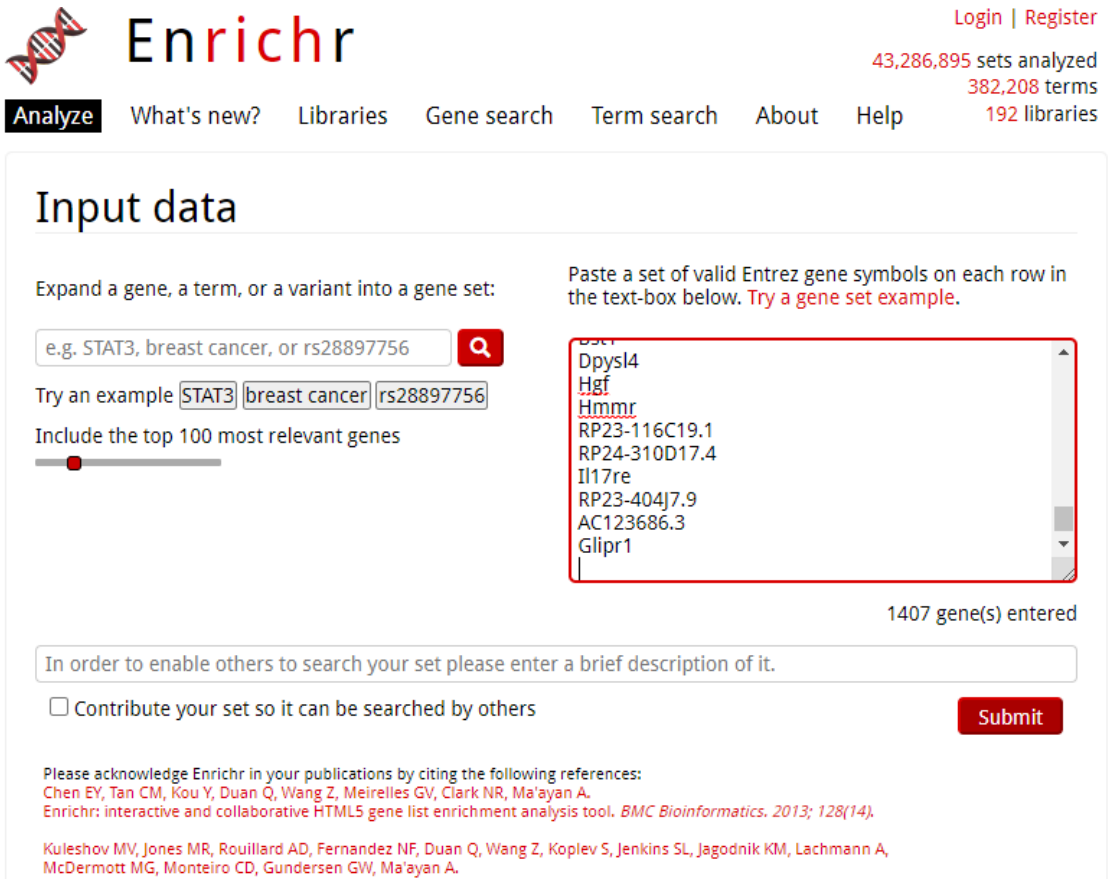
Script 3: Differential expression analysis with DESeq2

```
#Advanced - For loop that tests every comparison
for (i in unique(sample_info$timepoint)) {
  #get samples
  samples <- rownames(sample_info[sample_info$timepoint==i,])
  #make DEseq data object
  dds <- DESeqDataSetFromMatrix(countData = read_counts[,samples], colData = sample_info[samples,], design = ~treatment)
  dds <- DESeq(dds)

  #get differential expression results
  resultsNames(dds)
  res <- results(object = dds, contrast = c("treatment", "RML", "Mock"))

  #clean up results file
  res <- res[order(res$padj),]
  res <- na.omit(res)
  res <-as.data.frame(res)
  write.csv(res, paste0("DE results/RML_", i, "_DE_results.csv"))
  print(paste0("saving file... ", "DE results/RML_", i, "_DE_results.csv"))
}
```

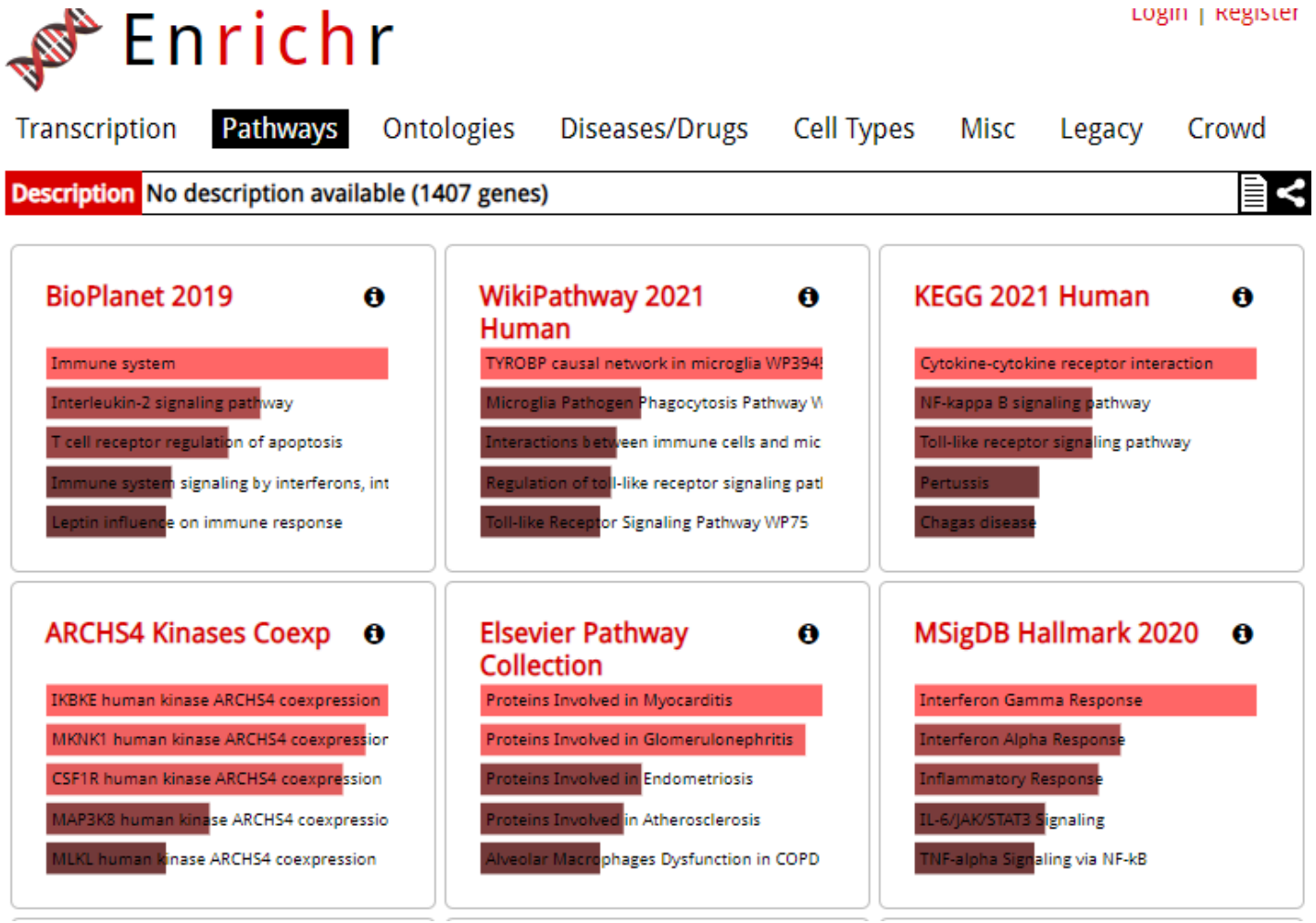
Script 4: Functional enrichment with enrichR

```r

library(enrichR)
library(dplyr)

#Identify DE genes
res <- read.csv("DE results/RML_terminal_DE_results.csv")

#Get some genes to test in enrichr
res %>% filter(padj < 0.05, log2FoldChange > 0.85, baseMean > 15)%>%
  pull(X) %>%
  writeClipboard()#copies to clipboard... paste at https://maayanlab.cloud/Enrichr/

```

Script 4: Functional enrichment with enrichR



Script 4: Functional enrichment with enrichR



Script 4: Functional enrichment with enrichR

```r
#Make list of databases you are interested in
dbs <- c("WikiPathway_2021_Human", "GO_Cellular_Component_2021", "PanglaoDB_Augmented_2021")

#Genes with increased abundance
genes <- res %>% filter(padj < 0.05, log2FoldChange > 0.85, baseMean > 15) %>% pull(X)

#Run Enrichr
enrch <- enrichr(genes, dbs)

#convert results from list to data frame
for (i in dbs) {
  enrch[[i]]$database <- i
}
enrch <- do.call(rbind, enrch)
```

Script 4: Functional enrichment with enrichR

	Term	Overlap	P.value	Adjusted.P.value
WikiPathway_2021_Human.1	TYROBP causal network in microglia WP3945	41/61	4.934306e-33	2.486890e-30
WikiPathway_2021_Human.2	Microglia Pathogen Phagocytosis Pathway WP3937	23/40	7.160984e-17	1.804568e-14
WikiPathway_2021_Human.3	Interactions between immune cells and microRNAs in tumor...	18/28	1.068639e-14	1.795314e-12
WikiPathway_2021_Human.4	Regulation of toll-like receptor signaling pathway WP1449	39/139	3.377230e-14	4.255310e-12
WikiPathway_2021_Human.5	Toll-like Receptor Signaling Pathway WP75	32/103	3.059690e-13	3.084168e-11
WikiPathway_2021_Human.6	Type I interferon induction and signaling during SARS-CoV-...	17/31	2.371525e-12	1.992081e-10
WikiPathway_2021_Human.7	Toll-like Receptor Signaling related to MyD88 WP3858	16/31	3.619922e-11	2.606344e-09
WikiPathway_2021_Human.8	miRNAs involvement in the immune response in sepsis WP4...	17/37	9.474346e-11	5.968838e-09
WikiPathway_2021_Human.9	Type II interferon signaling (IFNG) WP619	16/37	1.034479e-09	5.793081e-08
WikiPathway_2021_Human.10	SARS-CoV-2 innate immunity evasion and cell-specific imm...	21/66	2.218446e-09	1.118097e-07
WikiPathway_2021_Human.11	Fibrin Complement Receptor 3 Signaling Pathway WP4136	15/41	5.258592e-08	2.409391e-06

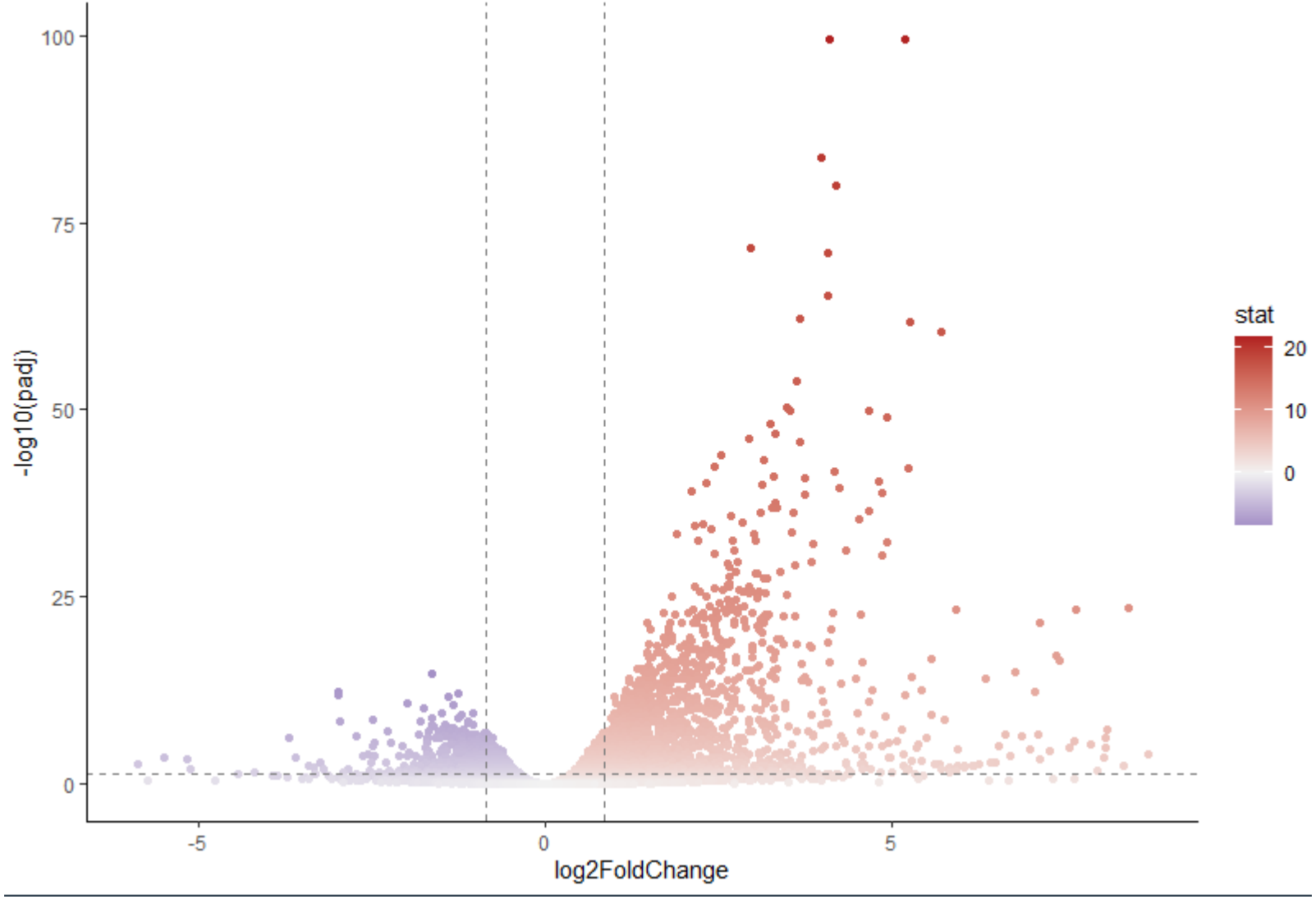
Script 4: Functional enrichment with enrichR

```r
#Make list of databases you are interested in
dbs <- c("WikiPathway_2021_Human", "GO_Cellular_Component_2021", "PanglaoDB_Augmented_2021")

#Make empty list for full results
full_enrch_results <- list()
#Loop through every list of genes
sample_info <- read.csv("sample_info.csv", row.names = 2)[,-1]
for (i in unique(sample_info$timepoint)) {
  res <- read.csv(paste0("DE results/RML_", i, "_DE_results.csv"))

  ##Genes with increased abundance
  genes <- res %>% filter(padj < 0.05, log2FoldChange > 0.85, baseMean > 15) %>% pull(X)
  if (length(genes) > 0) {
    enrch <- enrichr(genes, dbs)
    for (j in dbs) {
      enrch[[j]]$timepoint <- i
      enrch[[j]]$direction <- "up"
      enrch[[j]]$database <- j
    }
    enrch <- do.call(rbind, enrch)#convert from list to data frame
    full_enrch_results[[paste0(i, "_up")]] <- enrch
    print(paste0("analysis complete... ", i, "_up"))
  }
```

Script 5: Common data visualizations (volcano plot)

```r
library(ggplot2)
library(RColorBrewer)
library(pheatmap)
library(dplyr)

#The volcano plot
#Load differential expression results from terminal timepoint
res <- read.csv("DE results/RML_terminal_DE_results.csv")

#a basic volcano plot
ggplot(res, aes(x=log2FoldChange, y=-log10(padj))) +
  geom_point()

#a nicer volcano plot
ggplot(res, aes(x=log2FoldChange, y=-log10(padj), color=stat)) +
  geom_point() +
  geom_hline(yintercept = -log10(0.05), linetype="dashed", color="grey50") +
  geom_vline(xintercept = 0.85, linetype="dashed", color="grey50") +
  geom_vline(xintercept = -0.85, linetype="dashed", color="grey50") +
  scale_color_gradient2(low = "navy", high = "firebrick", mid="grey95", midpoint = 0) +
  theme_classic()
```

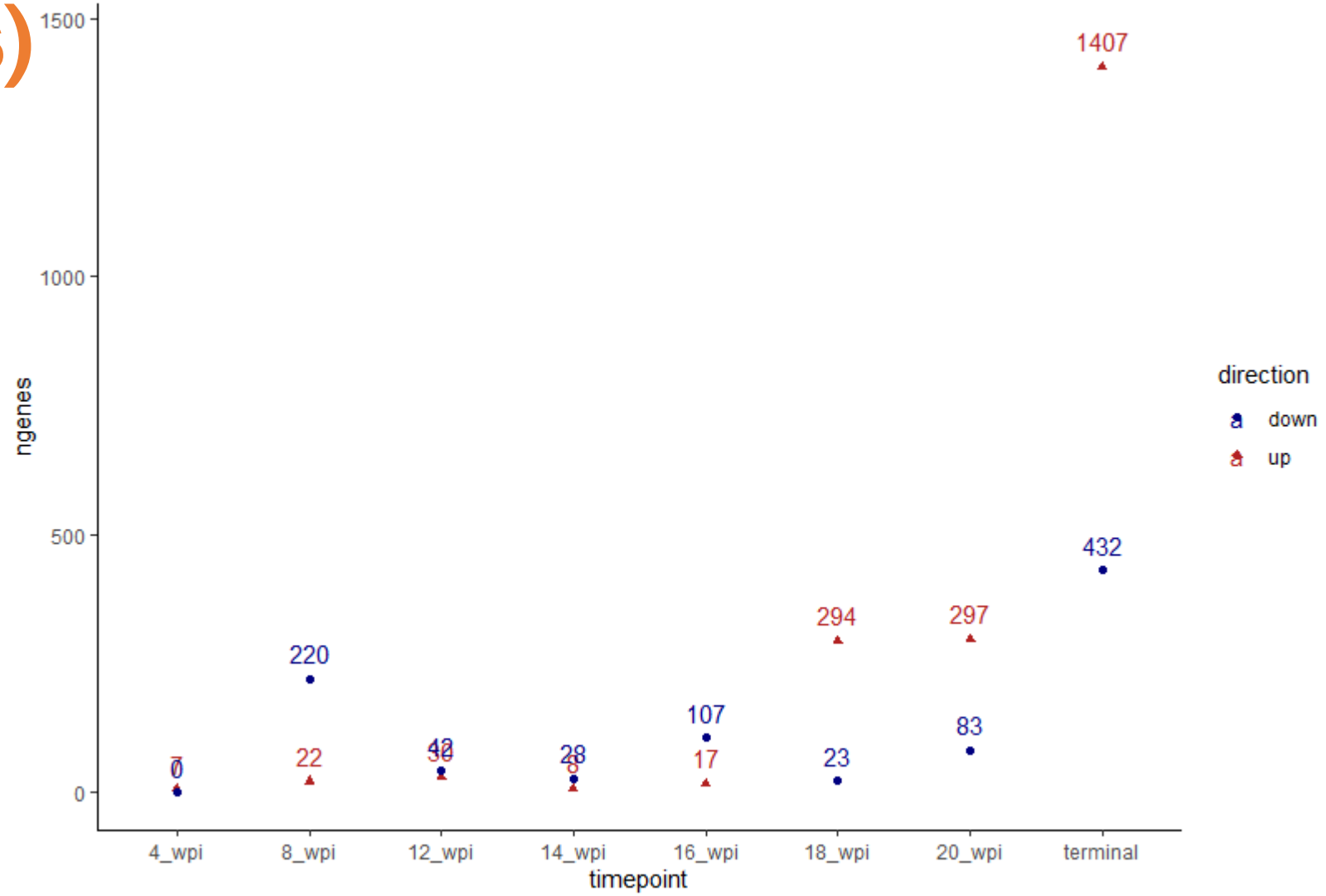
Script 5: Common data visualizations (volcano plot)



Script 5: Common data visualizations (plot n DE genes)

```
#Plotting number of DE genes at each timepoint
res <- data.frame(timepoint=factor(c("4_wpi","4_wpi","8_wpi","8_wpi","12_wpi","12_wpi","14_wpi","14_
                                     levels = c("4_wpi","8_wpi","12_wpi","14_wpi","16_wpi","18_wpi","2
                  direction=rep(c("up", "down"), 8),
                  ngenes=NA)
for (i in c("4_wpi","8_wpi","12_wpi","14_wpi","16_wpi","18_wpi","20_wpi","terminal")) {
  tmp <- read.csv(paste0("DE results/RML_", i, "_DE_results.csv"))
  res[res$timepoint==i&res$direction=="up",]$ngenes <- tmp %>% filter(padj < 0.05, log2FoldChange >
  res[res$timepoint==i&res$direction=="down",]$ngenes <- tmp %>% filter(padj < 0.05, log2FoldChange
  rm(tmp)
}

#a basic plot
ggplot(res, aes(x=timepoint, y=ngenes, color=direction, shape=direction)) +
  geom_point()

#a nicer plot
ggplot(res, aes(x=timepoint, y=ngenes, color=direction, shape=direction, label=ngenes)) +
  geom_point() +
  geom_text(nudge_y = 50) +
  scale_color_manual(values=c("navy", "firebrick")) +
  theme_classic()
```

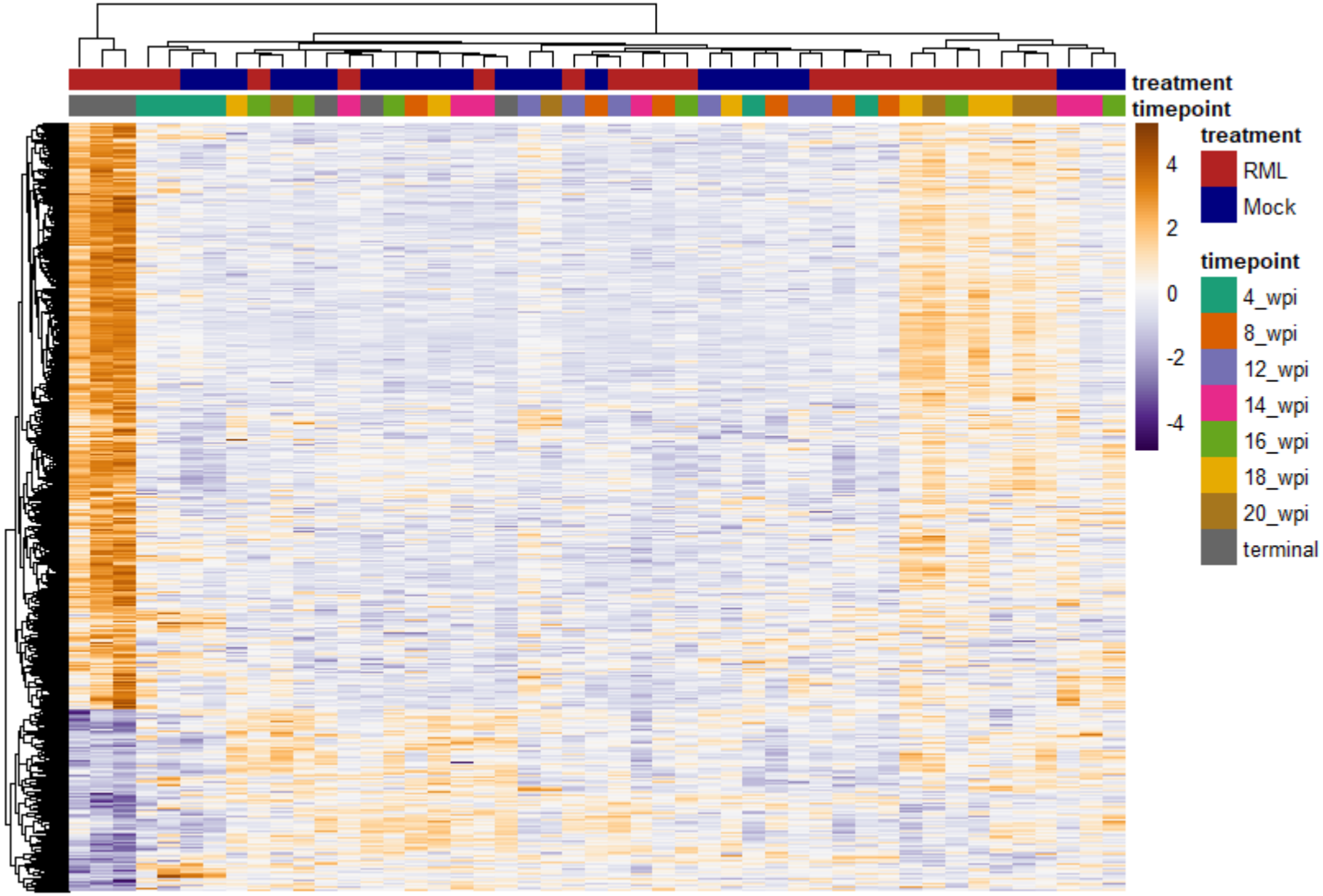
Script 5: Common data visualizations (plot n DE genes)



Script 5: Common data visualizations (heatmap)

```
#The heatmap
#We will use DE genes at terminal timepoint
genes <- read.csv("DE results/RML_terminal_DE_results.csv") %>%  filter(padj < 0.05, abs(log2FoldChange) > 0.85, baseMear

#We will use normalized read-counts to calculate z-scores
zscores <- read.csv("raw data/normalized_read_counts.csv", row.names = 1)
zscores <- as.matrix(zscores[genes,])
zscores <- (zscores-rowMeans(zscores))/matrixStats::rowSds(zscores)

#basic heatmap with hierarchical clustering
pheatmap(zscores)

#nicer heatmap
#specify additional variables required by pheatmap
plot_colors <- rev(colorRampPalette(brewer.pal(11,"PuOr"))(100))#colors for mapping to z-scores
column_annotation <- read.csv("sample_info.csv", row.names = 2)[,-1]#annotation for samples
cls <- brewer.pal(8, "Dark2")#colors for qualitative categorization of samples
annotation_colors <- list(`treatment`=c(`RML`="firebrick", `Mock`="navy"),#this list sets the colors for the annotation
                          `timepoint`=c(`4_wpi`=cls[1],`8_wpi`=cls[2],`12_wpi`=cls[3],`14_wpi`=cls[4],
                                        `16_wpi`=cls[5],`18_wpi`=cls[6],`20_wpi`=cls[7],`terminal`=cls[8]))

pheatmap(zscores, color = plot_colors, annotation_col = column_annotation, annotation_colors = annotation_colors,
         show_rownames = FALSE, show_colnames = FALSE, treeheight_row = 25, treeheight_col = 25)
```

Script 5: Common data visualizations (heatmap)



Script 5: Common data visualizations (plot enriched gene sets)

```r
#Plotting the enrichment results
erch <- read.csv("Enrichr results/full_enrichment_results.csv") #load full enrichment results
res <- erch %>% #filter to top 10 enriched WikiPathways increased at terminal timepoint
  filter(timepoint=="terminal", direction=="up", database == "WikiPathway_2021_Human") %>%
  arrange(Adjusted.P.value) %>%
  dplyr::slice(1:10)

#basic enrichment plot
ggplot(res, aes(x=-log10(Adjusted.P.value), y=Term)) +
  geom_point()

#nicer plot
#order Terms based on P-value by converting to a factor
res$Term <- factor(res$Term, levels = res$Term)
ggplot(res, aes(x=-log10(Adjusted.P.value), y=Term, color=Combined.Score, label=Overlap)) +
  geom_point(size=3, alpha=0.5) +
  geom_text(nudge_x = 0.5, hjust=0) +
  scale_color_gradientn(colors=brewer.pal(8, "Oranges")[3:8]) +
  theme_classic()
```

Script 5: Common data visualizations (plot enriched gene sets)

Summary

1. *Raw read count processing (Script 1)*
2. *Normalization with DESeq2 (Script 2)*
3. *Differential expression analysis with DESeq2 (Script 3)*
4. *Functional enrichment analysis with Enrichr (Script 4)*
5. *Common data visualizations (Script 5)*

HELPFUL RESOURCES

http://bioconductor.org/packages/devel/bioc/vignettes/DESeq2/inst/doc/DESeq2.html
https://amp.pharm.mssm.edu/Enrichr/
https://www.ncbi.nlm.nih.gov/pmc/articles/PMC6096346/

THANK YOU FOR ATTENDING!

The Q&A Session will now begin.

Please make sure to fill out the Exit Survey

We value your feedback!

More questions? Please email us at

mmid.coding.workshop@gmail.com or post them to the workshop slack channel