Papalexi Confounders

2023-02-21

Goal

The goal of this report is to identify which grna assignments may be confounded by biological replicate

```
# Load packages.
library(Seurat)
```

```
## Attaching SeuratObject
```

```
library(SeuratData)
```

```
## -- Installed datasets ------------------------------------- SeuratData v0.2.2 --

## v thp1.eccite 3.1.5

## -------------------------------------- Key -------------------------------------

## v Dataset loaded successfully
## > Dataset built with a newer version of Seurat than installed
## (?) Unknown version of Seurat installed
```

```
library(ggplot2)
library(patchwork)
library(scales)
library(dplyr)
```

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

## The following objects are masked from 'package:stats':
##
##     filter, lag

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

```
library(reshape2)
library(mixtools)
```

```
## mixtools package, version 2.0.0, Released 2022-12-04
## This package is based upon work supported by the National Science Foundation under Grant No. SES-0518
```

```
library(stringr)
library(ondisc)
library(sceptre2)
library(sceptre)
library(kableExtra)
```

```
##
## Attaching package: 'kableExtra'

## The following object is masked from 'package:dplyr':
##
##     group_rows
```

Read in Data

```
LOCAL_SCEPTRE2_DATA_DIR <-.get_config_path("LOCAL_SCEPTRE2_DATA_DIR")
papalexi_dir <- paste0(LOCAL_SCEPTRE2_DATA_DIR, "data/papalexi/eccite_screen/")

# gene info
gene_odm_fp <- paste0(papalexi_dir, "gene/matrix.odm")

# grna info
grna_odm_fp <- paste0(papalexi_dir, "grna_assignment/matrix.odm")

# protein info
protein_odm_fp <- paste0(papalexi_dir, "protein/matrix.odm")

# mm odm metadata fp
mm_metadata_fp <- paste0(papalexi_dir, "multimodal_metadata.rds")

# construct mm odm
mm_odm <- read_multimodal_odm(odm_fps = c(gene_odm_fp, grna_odm_fp,
                                          protein_odm_fp),
                              multimodal_metadata_fp = mm_metadata_fp)
```

Test if grna Assignment Differs By Replication

```
# get perturbations for gene
bio_rep = mm_odm@global_cell_covariates$bio_rep
grna = str_sub(mm_odm@modalities$grna_assignment@cell_covariates$assigned_grna,
               1,-3)
grna[grna == 'NTg'] = 'NT'

#get unique targets
targets = unique(grna)
targets = targets[targets != 'NT']
#initialize vector that will hold pvalues testing if there is an association
```

```r
#between bio_rep and grna assignment
confounder = rep(NA,length(targets))
names(confounder) = targets
#initialize counter
counter = 1
#iterate over grna targets
for(gene in targets){
  #get cells which are either NT or gene KO
  cells = which(grna == gene | grna == 'NT')
  #get grna assignments
  A = grna[cells]
  #get biological replicate
  B = bio_rep[cells]
  #get pvalue via fishers test
  confounder[counter] = fisher.test(x = table(B,A))$p.value
  #counter++
  counter = counter + 1
}
#BH correction
confounder = p.adjust(confounder,method = 'BH')
#see which grna assignments differ by replicate
affected_confounders = confounder[which(confounder<0.05)]
results = cbind(names(affected_confounders),affected_confounders)
colnames(results) = c('Perturbation','Pvalue')
rownames(results) = c(1:nrow(results))
results_table = kable(results,booktabs = TRUE, linesep = "")
kable_styling(results_table,position = "center", latex_options = "scale_down")
```

We see that CUL3 and BRD4 are the main grnas that are affected by biological replicate. I will now perform seuratDE analysis, stratifying by biological replicate. Recall that seuratDE finds that CUL3 and BRD4 KO increase the expression of PDL1 protein and mrna. Do these results hold true if we stratify by replicate?

Perturbation	Pvalue
STAT1	9.3653994759453e-08
CD86	0.00291638964136335
IRF7	2.18064264133147e-06
JAK2	0.0214561515007108
NFKBIA	1.8216433816787e-06
SMAD4	7.3218654846214e-12
IFNGR1	3.81342050449118e-08
UBE2L6	0.0214561515007108
PDCD1LG2	2.08510775019386e-06
CUL3	0.0214561515007108
BRD4	0.0214561515007108
MARCH8	0.000479350842010373
IRF1	2.86745558189527e-20
POU2F2	1.76392526733969e-07
SPI1	0.0313645041835894

SeuratDE Analysis Stratifying by Replicate

Loading Data

```
# Download dataset using SeuratData.
options(timeout = 1000)
InstallData(ds = "thp1.eccite")
```

```
## Warning: The following packages are already installed and will not be
## reinstalled: thp1.eccite
```

```
# Setup custom theme for plotting.
custom_theme <- theme(
  plot.title = element_text(size=16, hjust = 0.5),
  legend.key.size = unit(0.7, "cm"),
  legend.text = element_text(size = 14))

# Load object.
eccite <- LoadData(ds = "thp1.eccite")
```

Preprocessing

```
# Normalize protein.
eccite <- NormalizeData(
  object = eccite,
  assay = "ADT",
  normalization.method = "CLR",
  margin = 2)
```

```
## Normalizing across cells
```

```
# Prepare RNA assay for dimensionality reduction:
# Normalize data, find variable features and scale data.
DefaultAssay(object = eccite) <- 'RNA'
eccite <- NormalizeData(object = eccite) %>%
  FindVariableFeatures() %>%
  ScaleData()
```

```
## Centering and scaling data matrix
```

```
# Run Principle Component Analysis (PCA)
eccite <- RunPCA(object = eccite)
```

```
## PC_ 1
## Positive:  BIRC5, TOP2A, CDC20, MKI67, CENPF, TPX2, CDKN3, UBE2C, CKS1B, NUF2
##     CCNA2, NUSAP1, KIAA0101, CENPA, HMGB2, SGOL1, TYMS, STMN1, MYBL2, GTSE1
##     ASPM, H2AFZ, CDCA2, HMMR, CDCA8, KIF2C, CKAP2L, PTTG1, MND1, UBE2T
## Negative:  FTH1, FCER1G, NEAT1, SOD2, FTL, MAFB, BTG1, NPC2, CTSL, CTSC
##     CTSB, SLC31A2, CHI3L1, FAM26F, TNFSF13B, GBP5, PLAUR, EVL, GK, ASAH1
##     HLA-DRB1, HLA-DRA, SPP1, SCPEP1, CD74, SAT1, GBP1, SLAMF7, WARS, SDS
## PC_ 2
## Positive:  HYOU1, PDIA4, HSPA5, SDF2L1, MEI1, MANF, DNAJB9, NUCB2, TRIB3, WIPI1
##     CRELD2, HSP90B1, MSTO1, SLC39A14, HERPUD1, ALDH1L2, DERL3, VIMP, SEC11C, SERP1
##     PPAPDC1B, CDK2AP2, OSTC, DNAJB11, ERO1LB, SEC61G, SYVN1, TMED2, DNAJC3, PYCR1
## Negative:  HSPA8, KIAA0101, TYMS, MKI67, FCER1G, CHI3L1, ACTG1, TOP2A, MYBL2, HSP90AA1
##     CCNA2, BIRC5, CLSPN, PKMYT1, NPC2, NUSAP1, HMGN2, ZWINT, CENPF, H2AFZ
##     TMEM106C, CENPW, TUBA1B, STMN1, CTSC, ASF1B, CDCA5, HMGA1, RRM2, GTSE1
## PC_ 3
## Positive:  CDKN1A, ATF5, WARS, PLEK, CXCL10, IL1RN, SOD2, FAM26F, SLC31A2, GBP1
##     IDO1, SLAMF7, GK, HLA-DRA, ISG20, ICAM1, CD274, CCL2, ATF3, GBP5
##     CCL8, CD74, MTHFD2, IL8, FCER1G, GCH1, TNFSF13B, IL4I1, GLUL, RALA
## Negative:  QPRT, S100A4, RPLP0, S100A6, ZFP36L2, ALOX5AP, SORL1, ANTXR1, C1orf162, VCAN
##     GLIPR1, CD1D, ID1, CAPN2, ID2, TGFBR1, RGS16, TKT, ITM2C, CDKN2C
##     HSPB1, ACTG1, CORO1A, SMYD3, ID3, RPSA, ALDH2, FOS, AZU1, THYN1
## PC_ 4
## Positive:  RMDN3, GCHFR, GRN, DNASE2, WARS, SCCPDH, PSME2, LIPG, CTSD, HLA-DRB1
##     C19orf59, TSPO, HLA-DRB5, LTA4H, HLA-A, IFI30, AGT, GBP5, CEBPE, APOC1
##     GLUL, MARC1, CD74, CD1D, PPARG, ALOX5AP, CLDN23, CD68, S100A8, PLIN2
## Negative:  CCL2, IGFBP3, PEA15, CCL3, NFKBIA, MMP9, CCL4, CCL5, POU2F2, IL1B
##     MARCKSL1, CXCL11, MX2, RGS1, CXCL9, USP18, PDPN, SPP1, CLEC5A, E2F1
##     TGFBR1, CKB, RUNX3, PTPN14, SMYD3, TGFBI, CCL8, TESC, GINS2, PNRC1
## PC_ 5
## Positive:  FTL, FABP5, PLIN2, CSTB, CTSD, CD36, FTH1, HMGA1, SPOCD1, RMDN3
##     APOC1, DDIT4L, AGPAT9, CDK4, MSR1, E2F1, SRM, GCHFR, GINS2, CLDN23
##     TOMM40, RND3, NCF2, FAM111B, DTL, APOE, STRA13, IL8, SLC11A1, CHCHD10
## Negative:  TNFSF10, NCF1, PSMB9, CXCL11, CXCL10, IFI27, TMEM176B, RARRES3, TMEM176A, NFKBIA
##     IL32, SOCS1, IFITM1, GBP1, CCNB1, PLK1, HMMR, PNRC1, PSME2, RGS16
##     MYO1G, CD74, ISG20, TNFSF13B, CXCL9, TMEM50B, CDC20, IFIT2, PTTG1, GLIPR1
```

```r
# Run Uniform Manifold Approximation and Projection (UMAP)
eccite <- RunUMAP(object = eccite, dims = 1:40)
```

```
## Warning: The default method for RunUMAP has changed from calling Python UMAP via reticulate to the R
## To use Python UMAP via reticulate, set umap.method to 'umap-learn' and metric to 'correlation'
## This message will be shown once per session
```

```
## 08:07:06 UMAP embedding parameters a = 0.9922 b = 1.112
```

```
## 08:07:06 Read 20729 rows and found 40 numeric columns
```

```
## 08:07:06 Using Annoy for neighbor search, n_neighbors = 30
```

```
## 08:07:06 Building Annoy index with metric = cosine, n_trees = 50
```

```
## 0%   10   20   30   40   50   60   70   80   90   100%
```

```
## [----|----|----|----|----|----|----|----|----|----|
## **************************************************|
## 08:07:07 Writing NN index file to temp file /var/folders/tf/8hslpl416b70psyc3zvyzfxm0000gn/T//RtmpkF
## 08:07:07 Searching Annoy index using 1 thread, search_k = 3000
## 08:07:11 Annoy recall = 100%
## 08:07:11 Commencing smooth kNN distance calibration using 1 thread with target n_neighbors = 30
## 08:07:11 Initializing from normalized Laplacian + noise (using irlba)
## 08:07:12 Commencing optimization for 200 epochs, with 910832 positive edges
## 08:07:20 Optimization finished
```

Removing Technical Effects

```
# Calculate perturbation signature (PRTB).
eccite<- CalcPerturbSig(
  object = eccite,
  assay = "RNA",
  slot = "data",
  gd.class ="gene",
  nt.cell.class = "NT",
  reduction = "pca",
  ndims = 40,
  num.neighbors = 20,
  split.by = "replicate",
  new.assay.name = "PRTB")
```

```
## Processing rep1

## Processing rep3

## Processing rep2
```

```
# Prepare PRTB assay for dimensionality reduction:
# Normalize data, find variable features and center data.
DefaultAssay(object = eccite) <- 'PRTB'

# Use variable features from RNA assay.
VariableFeatures(object = eccite) <- VariableFeatures(object = eccite[["RNA"]])
eccite <- ScaleData(object = eccite, do.scale = F, do.center = T)
```

```
## Centering data matrix
```

```
# Run PCA to reduce the dimensionality of the data.
eccite <- RunPCA(object = eccite, reduction.key = 'prtbpca',
                 reduction.name = 'prtbpca')
```

```
## Warning: Keys should be one or more alphanumeric characters followed by an
## underscore, setting key from prtbpca to prtbpca_

## Warning: All keys should be one or more alphanumeric characters followed by an
## underscore '_', setting key to prtbpca_
## prtbpca_ 1
## Positive:  SPP1, S100A4, RPLP0, VCAN, ZFP36L1, TREM2, TGFBR1, CAPN2, TGFBI, LGALS1
##     RPSA, SORL1, FSCN1, CSF1R, YWHAH, LMNA, RPS2, ADORA3, HSPB1, CORO1A
##     ID2, MMP9, VAT1, GLO1, COL6A1, AP1S2, NFKBIA, MGST3, APOE, IL8
## Negative:  CD74, HLA-DRA, CXCL10, WARS, GBP5, GBP1, IFI27, HLA-DRB1, FAM26F, PSMB9
##     HLA-DRB5, IL18BP, PSME2, SOCS1, HLA-DPA1, HLA-DQB1, SOD2, IFITM1, NCF1, S100A8
##     HLA-A, GLUL, CTSL, CD70, FCGR1B, HLA-DMA, HLA-DPB1, FCER1G, LY6E, CHI3L1
## prtbpca_ 2
## Positive:  CXCL10, CXCL11, CXCL9, GBP1, SOCS1, GBP5, SOD2, TNFSF13B, CCL2, IFIT3
##     IL32, MX1, GYPC, IL18BP, ISG20, WARS, TNFSF10, IDO1, LY6E, IFI27
##     RSAD2, BAZ1A, FAM26F, IFIT2, GCH1, CD274, USP18, FTH1, TMEM176A, H1F0
## Negative:  S100A8, S100A9, ALOX5AP, S100A4, SPP1, CTSD, C19orf59, S100A6, GRN, APOC1
##     TREM2, CHI3L1, S100A10, ANXA2, GLO1, CALR, PPIB, TSPO, TIMP1, HLA-DRB5
##     IL8, SRGN, PLAUR, VIM, DNASE2, FABP5, LGALS1, HLA-DRB1, FN1, FCER1G
## prtbpca_ 3
## Positive:  CXCL10, CCL2, CXCL11, S100A9, S100A8, ALOX5AP, ISG15, CXCL9, IFI6, MX1
##     LY6E, CCL8, CYP1B1, S100A10, IL32, MARCKSL1, NFKBIA, GLO1, AP1S2, IFITM1
##     MAFB, IL8, LGALS1, S100A4, SOD2, CCL3, CDKN1A, SAT1, CCL5, IFIT3
## Negative:  HLA-DRA, CD74, HLA-DPA1, HLA-DRB5, HLA-DRB1, HLA-DQB1, HLA-DMA, HLA-DPB1, WARS, SPP1
##     PSME2, APOC1, ZFP36L2, CDKN2C, HLA-DQA1, SCPEP1, FAM26F, GBP1, IFI27, TKTL1
##     HLA-DMB, CD52, IL18BP, EVL, AZU1, FABP4, RARRES3, HLA-A, LGALS3, NCF1
## prtbpca_ 4
## Positive:  APOC1, HSP90B1, CALR, APOE, LGALS3, PDIA6, PPIB, HSPA5, PDIA4, PDIA3
##     P4HB, CXCL10, CRELD2, MANF, DNAJB11, CSTB, OSTC, C19orf10, PRDX1, SDF2L1
##     FABP5, CANX, CXCL11, CTSD, EMP3, KDELR2, HYOU1, CD68, ASAH1, SEC11C
## Negative:  S100A8, S100A9, S100A12, CYP1B1, IL8, ALOX5AP, NCF1, NCF2, LSP1, SLPI
##     CHI3L1, GLUL, CEBPE, SPP1, ZFP36L2, LTA4H, S100A6, LINC01094, SRPK1, AGPAT9
##     TKTL1, PLAC8, C19orf59, CXCL1, HMGB2, FOS, GBP5, C1orf162, FCER1G, CLDN23
## prtbpca_ 5
## Positive:  HSP90B1, S100A8, CALR, HSPA5, S100A9, PDIA4, PDIA6, PPIB, SDF2L1, CRELD2
##     MANF, PDIA3, P4HB, DNAJB11, C19orf10, HYOU1, HSPE1, OSTC, HSP90AB1, CANX
##     CXCL9, CCL2, SEC11C, S100A12, SRM, SEC61G, TMED2, KDELR2, HSPA8, DNAJC3
## Negative:  APOC1, NUPR1, FN1, APOE, SPP1, NEAT1, S100A10, LGALS3, BTG1, SAT1
##     JUN, VIM, DUSP1, FTH1, CSTB, PLIN2, MAFB, ISG15, PHLDA1, CTSD
##     SQSTM1, IFI6, CD70, GCHFR, CXCL10, TXNIP, NMB, FOS, ID2, GLUL
```

```r
# Run UMAP to visualize clustering in 2-D.
eccite <- RunUMAP(
  object = eccite,
  dims = 1:40,
  reduction = 'prtbpca',
  reduction.key = 'prtbumap',
  reduction.name = 'prtbumap')
```

```
## 08:07:36 UMAP embedding parameters a = 0.9922 b = 1.112
```

```
## 08:07:36 Read 20729 rows and found 40 numeric columns
```

```
## 08:07:36 Using Annoy for neighbor search, n_neighbors = 30
```

```
## 08:07:36 Building Annoy index with metric = cosine, n_trees = 50
```

```
## 0%   10   20   30   40   50   60   70   80   90   100%
## [----|----|----|----|----|----|----|----|----|----|
```

```
## **************************************************|
## 08:07:37 Writing NN index file to temp file /var/folders/tf/8hslpl416b70psyc3zvyzfxm0000gn/T//RtmpkF
## 08:07:37 Searching Annoy index using 1 thread, search_k = 3000
## 08:07:41 Annoy recall = 100%
## 08:07:41 Commencing smooth kNN distance calibration using 1 thread with target n_neighbors = 30
## 08:07:42 Initializing from normalized Laplacian + noise (using irlba)
## 08:07:42 Commencing optimization for 200 epochs, with 842656 positive edges
## 08:07:50 Optimization finished
```

```
## Warning: Keys should be one or more alphanumeric characters followed by an
## underscore, setting key from prtbumap to prtbumap_
```

```
## Warning: All keys should be one or more alphanumeric characters followed by an
## underscore '_', setting key to prtbumap_
```

Mixscape

```r
# Run mixscape.
eccite <- RunMixscape(
  object = eccite,
  assay = "PRTB",
  slot = "scale.data",
  labels = "gene",
  nt.class.name = "NT",
  min.de.genes = 5,
  iter.num = 10,
  de.assay = "RNA",
  verbose = F,
  prtb.type = "KO")
```

```
## Warning in FindMarkers.default(object = data.use, slot = data.slot, counts =
## counts, : No features pass logfc.threshold threshold; returning empty
## data.frame
```

```
## number of iterations= 95
## number of iterations= 187
## number of iterations= 172
## number of iterations= 18
## number of iterations= 6
## number of iterations= 18
## number of iterations= 11
## number of iterations= 11
## number of iterations= 59
## number of iterations= 43
## number of iterations= 42
## number of iterations= 19
## number of iterations= 12
## number of iterations= 12
## number of iterations= 23
## number of iterations= 19
## number of iterations= 19
## number of iterations= 51
## number of iterations= 51
## number of iterations= 51
## number of iterations= 36
## number of iterations= 26
## number of iterations= 25
## number of iterations= 20
## number of iterations= 12
## number of iterations= 12
## number of iterations= 17
## number of iterations= 15
## number of iterations= 14
## number of iterations= 13
## number of iterations= 73
## number of iterations= 46
## number of iterations= 41
```

```r
#get seurat biological replicates
bio_rep = eccite@meta.data$MULTI_classification.global
#get number of replicates
unique_bio_rep = unique(bio_rep)
N_rep = length(unique_bio_rep)
#initalize gene and pvalue matrix
CUL3_effect = matrix(NA,N_rep,2)
BRD4_effect = matrix(NA,N_rep,2)
CUL3_effect_protein = matrix(NA,N_rep,2)
BRD4_effect_protein = matrix(NA,N_rep,2)
#name rownames and column names of matrix
colnames(CUL3_effect) = c("pvalue","effect size")
rownames(CUL3_effect) = unique_bio_rep
colnames(CUL3_effect_protein) = c("pvalue","effect size")
rownames(CUL3_effect_protein) = unique_bio_rep
#name rownames and column names of matrix
colnames(BRD4_effect) = c("pvalue","effect size")
rownames(BRD4_effect) = unique_bio_rep
colnames(BRD4_effect_protein) = c("pvalue","effect size")
rownames(BRD4_effect_protein) = unique_bio_rep
#save counter
counter = 1
#perform seuratDE over each replicate
for (val in unique_bio_rep){
  #subset data
  data_sub = subset(eccite,MULTI_classification.global == val)
  #perform seuratDE
  CUL3_mrna = FindMarkers(data_sub,ident.1 = "CUL3 KO",ident.2 = 'NT',
                         features = 'CD274',assay = 'RNA',logfc.threshold = 0)
  BRD4_mrna = FindMarkers(data_sub,ident.1 = "BRD4 KO",ident.2 = 'NT',
                         features = 'CD274',assay = 'RNA',logfc.threshold = 0)

  CUL3_protein = FindMarkers(data_sub,ident.1 = "CUL3 KO",ident.2 = 'NT',
                         features = 'PDL1',assay = 'ADT',logfc.threshold = 0)
```

	pvalue	effect size
rep1-tx	0.0000019	0.7250975
rep3-tx	0.0041032	0.5331769
rep4-tx	0.0001197	0.7827858

```
  BRD4_protein = FindMarkers(data_sub,ident.1 = "BRD4 KO",ident.2 = 'NT',
                             features = 'PDL1',assay = 'ADT',logfc.threshold = 0)
  #save results
  CUL3_effect[counter,] = c(CUL3_mrna$p_val,CUL3_mrna$avg_log2FC)
  BRD4_effect[counter,] = c(BRD4_mrna$p_val,BRD4_mrna$avg_log2FC)
  CUL3_effect_protein[counter,] = c(CUL3_protein$p_val,CUL3_protein$avg_log2FC)
  BRD4_effect_protein[counter,] = c(BRD4_protein$p_val,BRD4_protein$avg_log2FC)
  #counter++
  counter = counter + 1

}
```

```
## For a more efficient implementation of the Wilcoxon Rank Sum Test,
## (default method for FindMarkers) please install the limma package
## --------------------------------------------
## install.packages('BiocManager')
## BiocManager::install('limma')
## --------------------------------------------
## After installation of limma, Seurat will automatically use the more
## efficient implementation (no further action necessary).
## This message will be shown once per session
```

CUL3 Results

PDL1 mRNA

Recall that seurat reports a pvalue of 1e-11 and an effect size of 0.7. It seems as though the effect size estimate is accurate despite some deviations in replication 3. Additionally, all pvalues are significant.

```
results_table = kable(CUL3_effect,booktabs = TRUE, linesep = "")
kable_styling(results_table,position = "center", latex_options = "scale_down")
```

	pvalue	effect size
rep1-tx	0.0000004	0.1489930
rep3-tx	0.0128913	0.1279574
rep4-tx	0.0001458	0.1878718

	pvalue	effect size
rep1-tx	0.0004061	0.4636287
rep3-tx	0.0091481	0.3671729
rep4-tx	0.3910406	0.2450499

PDL1 Protein

Recall that seurat reports a pvalue of 1e-11 and an effect size of 0.15. It seems as though the effect size estimate is accurate despite some deviations in replication 3. This again seems accurate despite all pvalues being significatnly greater than 1e-11. They are still significant however.

```
results_table = kable(CUL3_effect_protein,booktabs = TRUE, linesep = "")
kable_styling(results_table,position = "center", latex_options = "scale_down")
```

BRD4 Results

PDL1 mRNA

Seurat reports a pvalue of 1e-5 and a log fold change of 0.39. I note that in replication 4, the change is actually insignificant and no pvalue is less than 1e-5.

```
results_table = kable(BRD4_effect,booktabs = TRUE, linesep = "")
kable_styling(results_table,position = "center", latex_options = "scale_down")
```

	pvalue	effect size
rep1-tx	0.00e+00	0.1727459
rep3-tx	0.00e+00	0.2646587
rep4-tx	2.57e-05	0.2083611

PDL1 Protein

Seurat reports a pvalue of 1e-29 and a log fold change of 0.22. The results are invariant to replication.

```
results_table = kable(BRD4_effect_protein,booktabs = TRUE, linesep = "")
kable_styling(results_table,position = "center", latex_options = "scale_down")
```