Nasonia

Kimberly Olney & Heini Natri

06/11/2020

Loading libraries

```
# Install libraries for the first time using BiocManager:
# if (!requireNamespace("BiocManager", quietly = TRUE))
# install.packages("BiocManager")
# BiocManager::install()
# source("https://bioconductor.org/biocLite.R")
# BiocInstaller::biocLite("DESeq")

# load libraries
library(ggplot2)
library(ggExtra)
library(RColorBrewer)
library(ggplot2)
library(dplyr)
library(tidyverse)
library("knitr")
```

Setting the environment parameters.

```
# package knitr is required to set the working directory
require("knitr")
# set the working directory
opts_knit$set(root.dir = "~/Dropbox (ASU)/NASONIA_take3/ASE/ASEReadCounter/")
```

Importing count and pheno type data to create DGEList object

```
# working with Wilson or Clark data
#dataGroup = "Wilson"
dataGroup = "Clark"

# hybrid sample IDs for Wilson data
#GV <- c("014453", "014454", "014455")
#VG <- c("014450", "014451", "014452")

# hybrid sample IDs for Clark data
GV <- c("SRR2773799", "SRR2773797")
# SRR2773798 not included, failed QC
VG <- c("SRR2773794", "SRR2773795", "SRR2773796")
```

```r
# Full file name of each sample for all sample, just the GV hybirds and just the VG hybirds
#allfiles <- paste0(hybirds, "_ASEReadCounter_minDepth30_HET_FD_GFF_ratio.bed")
GV_Nvit <-
  paste0("NvitRef/",
         GV,
         "_ASEReadCounter_minDepth30_HET_FD_GFF_ratio.bed")
VG_Nvit <-
  paste0("NvitRef/",
         VG,
         "_ASEReadCounter_minDepth30_HET_FD_GFF_ratio.bed")

# For Ngir reference genome, files end with _VgirRef
GV_Ngir <-
  paste0("NgirRef/",
         GV,
         "_ASEReadCounter_minDepth30_HET_FD_GFF_ratio_VgirRef.bed")
VG_Ngir <-
  paste0("NgirRef/",
         VG,
         "_ASEReadCounter_minDepth30_HET_FD_GFF_ratio_VgirRef.bed")

GV_files <- c(GV_Ngir, GV_Nvit)
VG_files <- c(VG_Ngir, VG_Nvit)

# set colors for the imbreed lines and the hybirds
viralPalette <- brewer.pal(8, "Set2")
#VV_Color <- viralPalette[1]
#GG_Color <- viralPalette[2]
VG_Color <- viralPalette[3]
GV_Color <- viralPalette[4]

#-------
# GV ASE data
#-------
GV_lists <- lapply(GV_files, function(x) {
  ASElist <- read.csv(x, header = T, sep = "\t")
  ASElist$chrPos <- paste(ASElist$contig, ASElist$position)
  return(ASElist)
})

GV_sharedSites <-
  map(GV_lists, ~ .$chrPos) %>% #creates a list just of the column of interest
  reduce(intersect)  #applies the intersect function cumulatively to all the lists shared in ASE

# filter GV ASE files to only include snps that are shared among replicates
GV_sharedSites_ASE = lapply(GV_files, function(x) {
  ASElist <- read.csv(x, header = T, sep = "\t")
  ASElist$chrPos <- paste(ASElist$contig, ASElist$position)
  ASElist <-
    ASElist[ASElist$chrPos %in% GV_sharedSites, ] # only include sites shared among all samples
  return(ASElist)
})
```

```r
# save GV_sharedSites_ASE as data frame
GV_df <- data.frame(GV_sharedSites_ASE) #GV_sharedSites_ASE

# filter to only include genes with at least 2 informative snps
GV_df <- GV_df[ave(1:nrow(GV_df), GV_df$geneID, FUN = length) > 1 ,]

colnames = c("ref_count")
GV_df$ref_avg <-
  sapply(colnames, function(x)
    rowMeans(GV_df [, grep(x, names(GV_df))]))

colnames = c("alt_count")
GV_df$alt_avg <-
  sapply(colnames, function(x)
    rowMeans(GV_df [, grep(x, names(GV_df))]))

colnames = c("total")
GV_df$total_avg <-
  sapply(colnames, function(x)
    rowMeans(GV_df [, grep(x, names(GV_df))]))

GV_ref <- GV_df %>% group_by(geneID) %>%
  summarise(mean = map(reduce(ref_avg, `map2`, `+`), `/`, n()))
names(GV_ref)[names(GV_ref) == "mean"] <- "GV_ref_avg"

GV_alt <- GV_df %>% group_by(geneID) %>%
  summarise(mean = map(reduce(alt_avg, `map2`, `+`), `/`, n()))
names(GV_alt)[names(GV_alt) == "mean"] <- "GV_alt_avg"

GV_total <- GV_df %>% group_by(geneID) %>%
  summarise(mean = map(reduce(total_avg, `map2`, `+`), `/`, n()))
names(GV_total)[names(GV_total) == "mean"] <- "GV_total_avg"

# merge GV_ref and GV_total by matching geneIDs
GV_merge <-
  merge(
    merge(GV_ref, GV_alt, by.x = "geneID", by.y = "geneID"),
    GV_total,
    by.x = "geneID",
    by.y = "geneID"
  )
GV_merge$geneID <- as.character(GV_merge$geneID)
GV_merge$GV_ref_avg <- as.numeric(GV_merge$GV_ref_avg)
GV_merge$GV_alt_avg <- as.numeric(GV_merge$GV_alt_avg)
GV_merge$GV_total_avg <- as.numeric(GV_merge$GV_total_avg)
GV_merge$GV_ratio <- GV_merge$GV_ref_avg / GV_merge$GV_total_avg
write.table(
  GV_merge,
  file = paste0("avgRefGenomes/", dataGroup, "_GV_ASE_minDepth30.txt"),
  sep = "\t",
  row.names = FALSE,
  quote = FALSE
)
```

```
#-------
# VG ASE data
#-------
VG_lists <- lapply(VG_files, function(x) {
  ASElist <- read.csv(x, header = T, sep = "\t")
  ASElist$chrPos <- paste(ASElist$contig, ASElist$position)
  return(ASElist)
})

VG_sharedSites <-
  map(VG_lists, ~ .$chrPos) %>% #creates a list just of the column of interest
  reduce(intersect)  #applies the intersect function cumulatively to all the lists shared in ASE

# filter VG ASE files to only include snps that are shared among replicates
VG_sharedSites_ASE = lapply(VG_files, function(x) {
  ASElist <- read.csv(x, header = T, sep = "\t")
  ASElist$chrPos <- paste(ASElist$contig, ASElist$position)
  ASElist <-
    ASElist[ASElist$chrPos %in% VG_sharedSites, ] # only include sites shared among all samples
  return(ASElist)
})

# save VG_sharedSites_ASE as data frame
VG_df <- data.frame(VG_sharedSites_ASE)

# filter to only include genes with at least 2 informative snps
VG_df <- VG_df[ave(1:nrow(VG_df), VG_df$geneID, FUN = length) > 1 ,]

colnames = c("ref_count")
VG_df$ref_avg <-
  sapply(colnames, function(x)
    rowMeans(VG_df [, grep(x, names(VG_df))]))

colnames = c("alt_count")
VG_df$alt_avg <-
  sapply(colnames, function(x)
    rowMeans(VG_df [, grep(x, names(VG_df))]))

colnames = c("total")
VG_df$total_avg <-
  sapply(colnames, function(x)
    rowMeans(VG_df [, grep(x, names(VG_df))]))

VG_ref <- VG_df %>% group_by(geneID) %>%
  summarise(mean = map(reduce(ref_avg, `map2`, `+`), `/`, n()))
names(VG_ref)[names(VG_ref) == "mean"] <- "VG_ref_avg"

VG_alt <- VG_df %>% group_by(geneID) %>%
  summarise(mean = map(reduce(alt_avg, `map2`, `+`), `/`, n()))
names(VG_alt)[names(VG_alt) == "mean"] <- "VG_alt_avg"

VG_total <- VG_df %>% group_by(geneID) %>%
  summarise(mean = map(reduce(total_avg, `map2`, `+`), `/`, n()))
```

```r
names(VG_total)[names(VG_total) == "mean"] <- "VG_total_avg"

# merge VG_ref and VG_total by matching geneIDs
VG_merge <-
  merge(
    merge(VG_ref, VG_alt, by.x = "geneID", by.y = "geneID"),
    VG_total,
    by.x = "geneID",
    by.y = "geneID"
  )
VG_merge$geneID <- as.character(VG_merge$geneID)
VG_merge$VG_ref_avg <- as.numeric(VG_merge$VG_ref_avg)
VG_merge$VG_alt_avg <- as.numeric(VG_merge$VG_alt_avg)
VG_merge$VG_total_avg <- as.numeric(VG_merge$VG_total_avg)
VG_merge$VG_ratio <- VG_merge$VG_ref_avg / VG_merge$VG_total_avg
write.table(
  VG_merge,
  file = paste0("avgRefGenomes/", dataGroup, "_VG_ASE_minDepth30.txt"),
  sep = "\t",
  row.names = FALSE,
  quote = FALSE
)

#--------
# intersect genes that are shared between
# GV and VG
#--------

sharedSites <- intersect(GV_sharedSites, VG_sharedSites)
write.table(
  sharedSites,
  file = paste0("avgRefGenomes/", dataGroup, "_GV&VG_sharedSites.txt"),
  sep = "\t",
  row.names = FALSE,
  quote = FALSE
)
sharedGenes <- intersect(GV_merge$geneID, VG_merge$geneID)
GV <- GV_merge[GV_merge$geneID %in% sharedGenes, ]
VG <- VG_merge[VG_merge$geneID %in% sharedGenes, ]
# remove geneID column
VG <- VG[-1]

# combine GV and VG data
ASE <- cbind(GV, VG)

#--------
# Fisher's exact test
#    test to determine if the allele bias in VG is the same
#    as the allele bias in GV
#--------
ASE$sig <- NA

for (i in 1:nrow(ASE)) {
```

```
  x2 <- as.integer(ASE[i, "VG_ref_avg"])# GV
  x4 <- as.integer(ASE[i, "GV_ref_avg"]) #VG
  x1 <- as.integer(ASE[i, "VG_alt_avg"])# GV
  x3 <- as.integer(ASE[i, "GV_alt_avg"]) #VG
  ASE_MAT <- matrix(c(x1, x2, x3, x4),
                    nrow = 2,
                    dimnames = list(Guess = c("x2", "x4"),
                                    Truth = c("x1", "x3")))
  test <- fisher.test(ASE_MAT, alternative = "two.sided")
  test$p.value
  ASE[i, "sig"] <- test$p.value
}

SIG <- subset(ASE, sig <= 0.05)
dim(SIG)
```

```
## [1] 18 10
```

```
ASE$FDRp <- p.adjust(ASE$sig, n = length(ASE$sig), method = "fdr")
sigFDR <- subset(ASE, FDRp <= 0.05)
dim(sigFDR)
```

```
## [1]  3 11
```

```
write.table(
  ASE,
  file = paste0("avgRefGenomes/", dataGroup, "_ASE_fisherExact.txt"),
  sep = "\t",
  row.names = FALSE,
  quote = FALSE
)

p <- ggplot(ASE) +
  geom_point(
    data = ASE,
    aes(x = VG_ratio, y = GV_ratio),
    alpha = 0.5,
    size = 1,
    color = "gray70"
  ) +
  theme(legend.position = "none") +
  theme_bw() +
  xlim(c(-0, 1)) + ylim(c(0, 1)) +
  scale_x_continuous(limits = c(0, 1), expand = c(0, 0)) +
  scale_y_continuous(limits = c(0, 1), expand = c(0, 0)) +
  labs(x = "VG", y = "GV") +
  geom_vline(
    xintercept = .6,
    linetype = "dotted",
    color = "lightpink3",
    size = .9
  ) +
  geom_hline(
    yintercept = .4,
    linetype = "dotted",
```

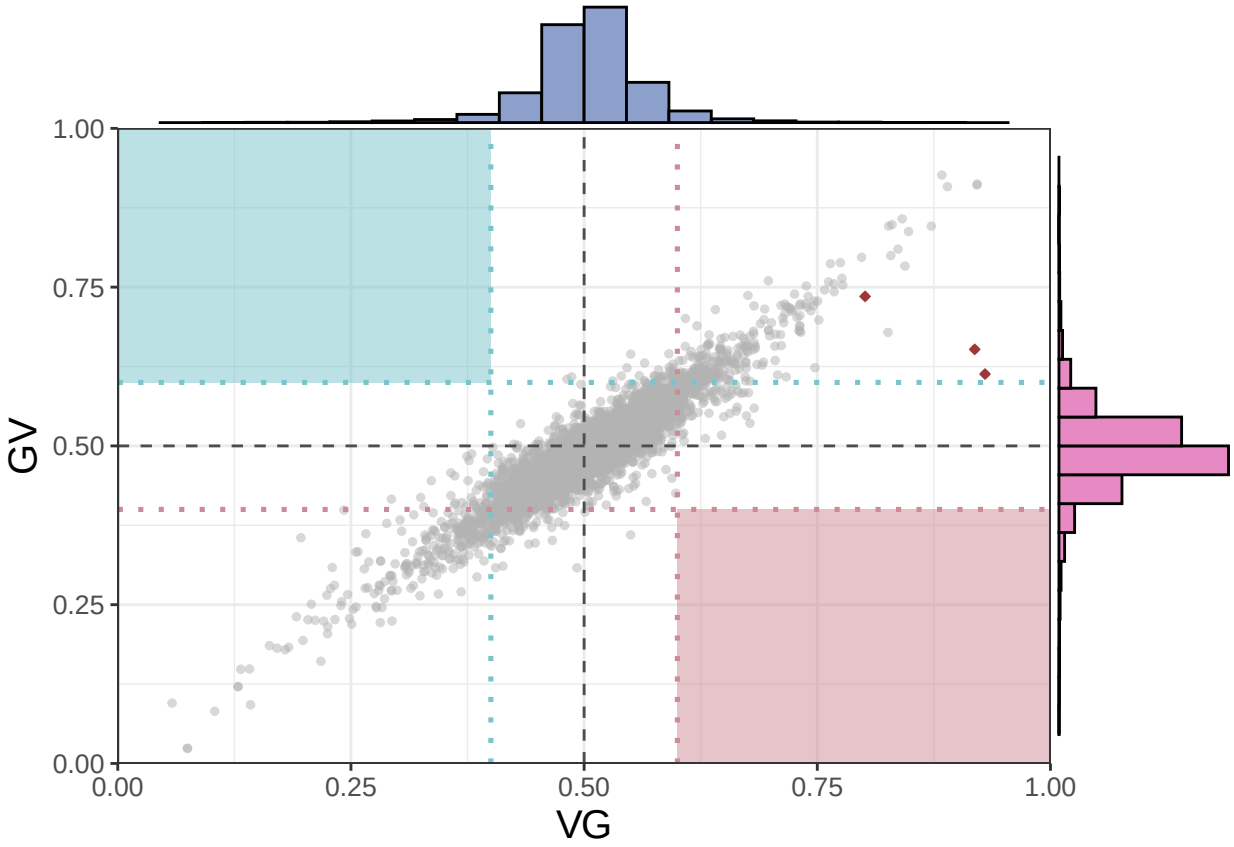
```
    color = "lightpink3",
    size = .9
  ) +
  geom_vline(
    xintercept = .4,
    linetype = "dotted",
    color = "cadetblue3",
    size = .9
  ) +
  geom_hline(
    yintercept = .6,
    linetype = "dotted",
    color = "cadetblue3",
    size = .9
  ) +
  geom_vline(
    xintercept = .5,
    linetype = "dashed",
    color = "gray30",
    size = .5
  ) +
  geom_hline(
    yintercept = .5,
    linetype = "dashed",
    color = "gray30",
    size = .5
  ) +
  scale_colour_manual(values = c(VG_Color, GV_Color)) +
  theme(axis.title.x = element_text(size = 15),
        axis.text.x = element_text(size = 10)) +
  theme(axis.title.y = element_text(size = 15),
        axis.text.y = element_text(size = 10))
```

```
## Scale for 'x' is already present. Adding another scale for 'x', which will
## replace the existing scale.
```

```
## Scale for 'y' is already present. Adding another scale for 'y', which will
## replace the existing scale.
```

```
p2 <-
  p +  annotate(
    "rect",
    xmin = 0.6,
    xmax = 1,
    ymin = 0,
    ymax = 0.4,
    fill = "lightpink3",
    alpha = .5
  )
p3 <-
  p2 +  annotate(
    "rect",
    xmin = 0,
    xmax = 0.4,
    ymin = .6,
```

```
    ymax = 1,
    fill = "cadetblue3",
    alpha = .5
  )
p4 <-
  p3 +  geom_point(
    data = sigFDR,
    aes(x = VG_ratio, y = GV_ratio),
    alpha = 0.75,
    size = 1.8,
    color = "darkred",
    shape = 18
  )
ggExtra::ggMarginal(
  p4,
  type = "histogram",
  binwidth = .05,
  fill = GV_Color,
  xparams = list(fill = VG_Color)
)
```

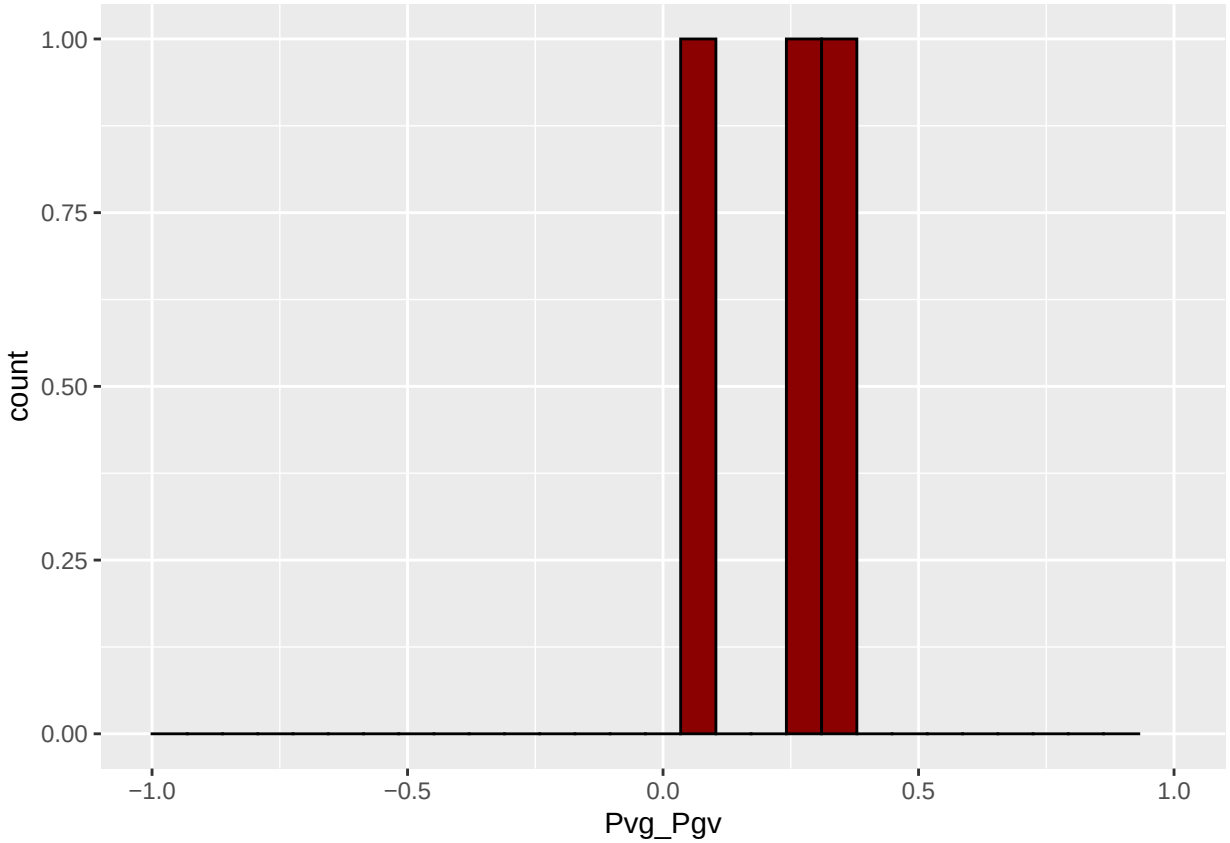



```
# dev.copy(
#   jpeg,
#   filename = paste0(
#     "avgRefGenomes/",
#     dataGroup,
#     "_ASE_fisherExact_minDepth30.jpg"
```

```
#   ),
#   width = 8,
#   height = 6,
#   units = "in",
#   res = 1200
# )

#dev.off()
#dev.off()
#----

sigFDR$Pvg_Pgv <- (sigFDR$VG_ratio - sigFDR$GV_ratio)
ASE <- subset(ASE, FDRp > 0.05)
ASE$Pvg_Pgv <- (ASE$VG_ratio - ASE$GV_ratio)

ggplot(sigFDR, aes(x = Pvg_Pgv)) + geom_histogram(color = "black", fill =
                                                    "darkred") +  xlim(c(-1, 1))
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```

```
## Warning: Removed 2 rows containing missing values (geom_bar).
```

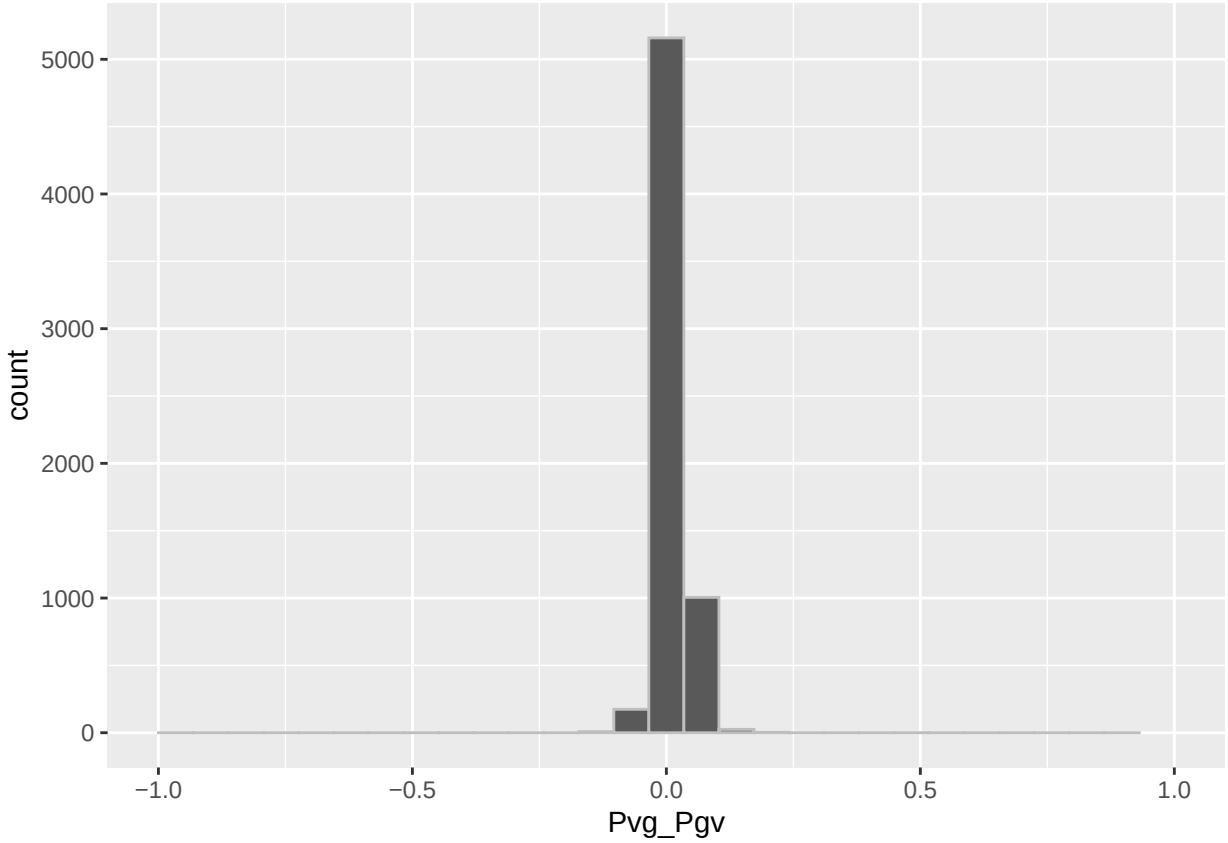



```
# dev.copy(
#   jpeg,
#   filename = paste0(
#     "avgRefGenomes/",
#     dataGroup,
#     "_ASE_fisherExact_histogram_FDR05.jpg"
```

```
#   ),
#   width = 8,
#   height = 6,
#   units = "in",
#   res = 1200
# )

#dev.off()
#dev.off()

ggplot(ASE, aes(x = Pvg_Pgv)) + geom_histogram(color = "gray") +  xlim(c(-1, 1))
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```

```
## Warning: Removed 2 rows containing missing values (geom_bar).
```




```
# dev.copy(
#   jpeg,
#   filename = paste0(
#     "avgRefGenomes/",
#     dataGroup,
#     "_ASE_fisherExact_histogram_excludeSIG.jpg"
#   ),
#   width = 8,
#   height = 6,
#   units = "in",
#   res = 1200
```

```
# )

#dev.off()
#dev.off()
```