Laura Biggins

4 March 2019

library("devtools")  
ifelse(using\_laptop,  
 load\_all("C:/Users/bigginsl/Desktop/temp/GOcategoryStats"),  
 load\_all("M:/GOcategoryStats")  
 )

## [[1]]  
## <environment: namespace:GOcategoryStats>

par(mgp = c(2, 0.5, 0))  
par(mar = par()$mar \* 0.7)  
  
library(RColorBrewer)  
palette(brewer.pal(6, "Set2"))

Generate lists of mouse genes biased by the length of the genes.

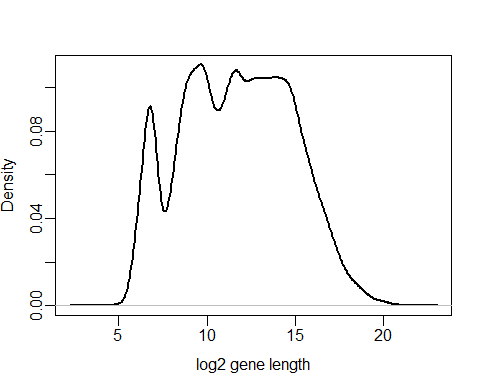
### Import the gene info file

The gene info file contains the lengths of all the genes in the mouse genome. These can be plotted so that some thresholds for length categories can be determined.

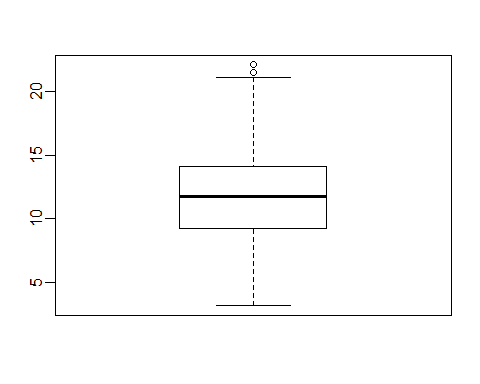
### Lengths of genes

#### Plot out the length distribution for all the genes in Mus\_musculus.GRCm38.94

x <- ifelse(using\_laptop,  
 genfo <- read.delim("C:/Users/bigginsl/Desktop/temp//biased\_gene\_lists/Mus\_musculus.GRCm38.94\_gene\_info.txt"),  
 genfo <- read.delim("M:/biased\_gene\_lists/Mus\_musculus.GRCm38.94\_gene\_info.txt")  
)  
bg\_genes <- as.vector(unique(genfo$gene\_name))  
  
plot(density(log2(genfo$length)), lwd = 2, main = "", xlab = "log2 gene length")



boxplot(log2(genfo$length))

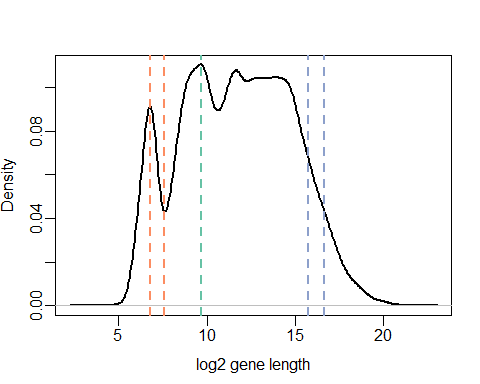


This data is displayed on a log scale as there are some very long genes that distort the graph if it is plotted on a linear scale. The plot shows a set of short genes between 32 and 194.0117205 (~5-7.5 on log2 scale). This provides a natural set of lengths to use, and can be split into a “very short” and a “short1” category. We’ll also take some of the genes that are slightly longer but are below the next peak in the distribution. As the distribution of lengths for the long genes is smooth, this provides no obvious thresholds to select, so we can take the top 5% and 10% of lengths.

long <- quantile(log2(genfo$length), probs = c(0.9, 0.95))  
names(long) <- c("long", "very\_long")

#### Select gene length thresholds.

plot(density(log2(genfo$length)),   
 lwd = 2,   
 main = "",   
 xlab = "log2 gene length"  
)  
  
thresholds <- c(  
 very\_short = 6.8,   
 short1 = 7.6,   
 short2 = 9.7,   
 long  
)  
  
more\_less <- c(  
 very\_short = "less",   
 short1 = "less",   
 short2 = "interval",   
 long = "more",   
 very\_long = "more"  
)  
  
sapply(names(thresholds), function(x){  
 colours <- as.factor(more\_less)  
 abline(v = thresholds[x], col = colours[x], lwd = 2, lty = 2)  
})



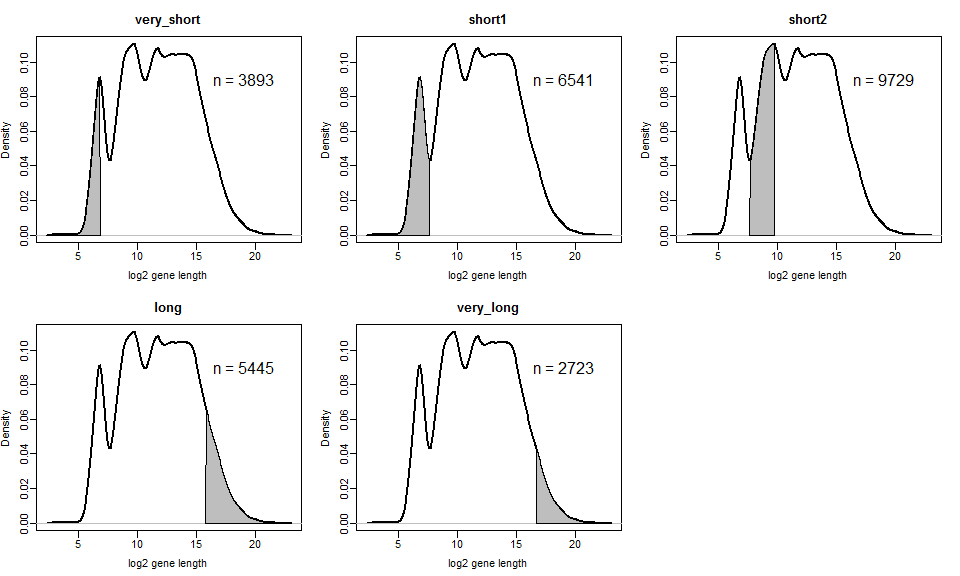
The thresholds

thresholds

## very\_short short1 short2 long very\_long   
## 6.80000 7.60000 9.70000 15.75344 16.66262

The number of genes that remain after filtering for each length category.

dens <- density(log2(genfo$length))  
  
par(mfrow = c(2, 3))  
  
sapply(names(thresholds), function(length\_cat){  
  
 plot(dens, lwd = 2, xlab = "log2 gene length", main = length\_cat)  
   
 if (more\_less[length\_cat] == "less") {  
 filt <- dens$x < thresholds[length\_cat]  
 polygon(  
 c(dens$x[filt], thresholds[length\_cat]),   
 c(dens$y[filt], 0),   
 col = "grey"  
 )  
 n\_genes <- sum(log2(genfo$length) < thresholds[length\_cat])  
   
 } else if (more\_less[length\_cat] == "more") {  
 filt <- dens$x > thresholds[length\_cat]  
 polygon(  
 c(dens$x[filt], thresholds[length\_cat]),   
 c(dens$y[filt], 0),   
 col = "grey"  
 )  
 n\_genes <- sum(log2(genfo$length) > thresholds[length\_cat])  
   
 } else {  
 # hard code the interval plot  
 filt <- dens$x > thresholds["short1"] & dens$x < thresholds["short2"]  
 polygon(  
 c(thresholds["short1"], dens$x[filt], thresholds["short2"]),   
 c(0, dens$y[filt], 0),   
 col = "grey"  
 )  
 n\_genes <- sum(log2(genfo$length) > thresholds["short1"] & log2(genfo$length) < thresholds["short2"] )  
 }  
   
 label\_text <- paste0("n = ", n\_genes)  
 text(x = 19, y = 0.09, labels = label\_text, cex = 1.5)  
})



### Further pre-processing: Generating the biased gene lists

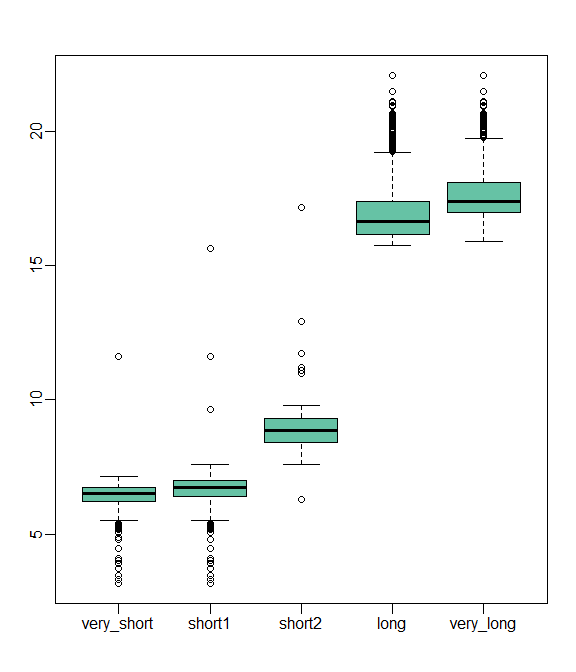
### Generating the biased gene lists within R

biased\_lengths <- lapply(names(thresholds), function(category){  
  
 if (more\_less[category] == "less") {  
 filtered\_genes <- genfo$gene\_name[log2(genfo$length) < thresholds[category]]  
 }  
 else if (more\_less[category] == "more") {  
 filtered\_genes <- genfo$gene\_name[log2(genfo$length) > thresholds[category]]  
 }  
 else {  
 # hard code the interval plot  
 filtered\_genes <- genfo$gene\_name[log2(genfo$length) > thresholds["short1"] &   
 log2(genfo$length) < thresholds["short2"]]  
 }  
 sapply(1:100, function(i){  
 filtered\_genes[ceiling(runif(200, min = 0, max = length(filtered\_genes) - 1))]  
 })  
})   
  
names(biased\_lengths) <- names(thresholds)

save(biased\_lengths, file = "M:/temp/length/length\_genelists.rda")

Check they all look right

par(mfrow = c(1, 1))  
  
lengths <- lapply(biased\_lengths, function(x) {  
 log2(genfo$length[match(unlist(x), genfo$gene\_name)])  
})  
  
boxplot(lengths, col = 1)



The odd gene that doesn’t fall within the expected size range may be due to name duplication.

### GO overrepresentation analysis

Run the gene lists through a GO overrepresentation analysis.

gene\_length\_results\_long <- lapply(biased\_lengths[4:5], function(length\_subset){  
   
 lapply(length\_subset, function(query){  
 overrep\_test(all\_go\_categories, query, bg\_genes)#, mult\_test = FALSE)  
 })  
})  
  
save(gene\_length\_results, file = "M:/GOcategoryStats/data/gene\_length\_results.rda")

We don’t want to run the GO analysis each time the document is knitted as it takes too long. The code was run once and the data saved as an .rda object that can be quickly loaded in to the R session.

See how many significant categories were returned.

ifelse(using\_laptop,  
 load("C:/Users/bigginsl/Desktop/temp//GOcategoryStats/data/length\_results.rda"),  
 load("M:/GOcategoryStats/data/length\_results.rda")  
)

## [1] "length\_results"

number\_of\_results <- lapply(length\_results, function(x){  
 nulls\_removed <- x[lapply(x,length) != 0]  
 vapply(nulls\_removed, nrow, FUN.VALUE = numeric(1))  
})

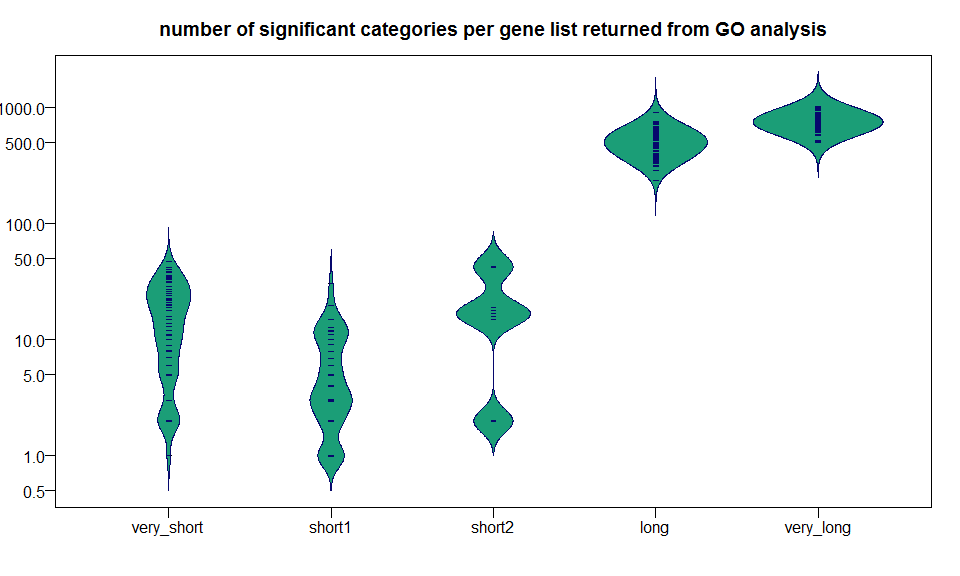
The number of gene sets that returned significant results from the GO overrepresentation analysis

sapply(number\_of\_results, length)

## very\_short short1 short2 long very\_long   
## 91 48 12 100 100

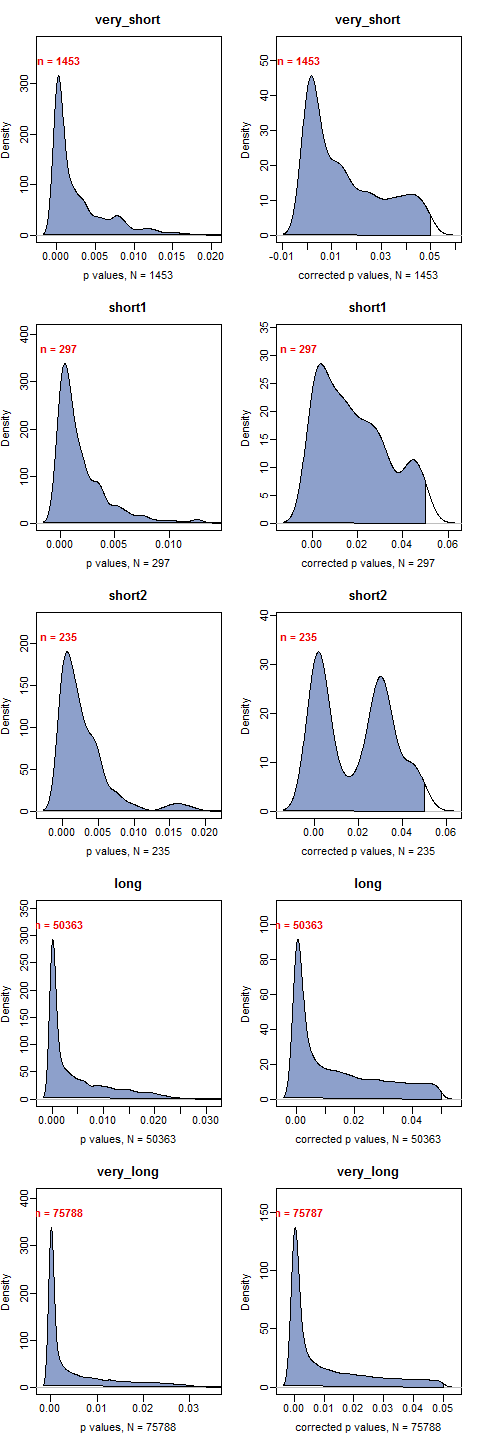
The bean plots show the number of categories returned from each gene list. The null results have been removed.

par(mfrow = c(1,1))  
  
library(beanplot)  
options(scipen = 999) # disable the scientific notation  
  
beanplot(  
 number\_of\_results,   
 what = c(0,1,0,1),   
 col = c("#1B9E77","#06086d"),   
 ll = 0.03,   
 method = "jitter",   
 border = "#06086d",  
 las = 1,  
 main = "number of significant categories per gene list returned from GO analysis"  
)



There are loads of results returned for the long genes, we’ll plot out the p and q values.

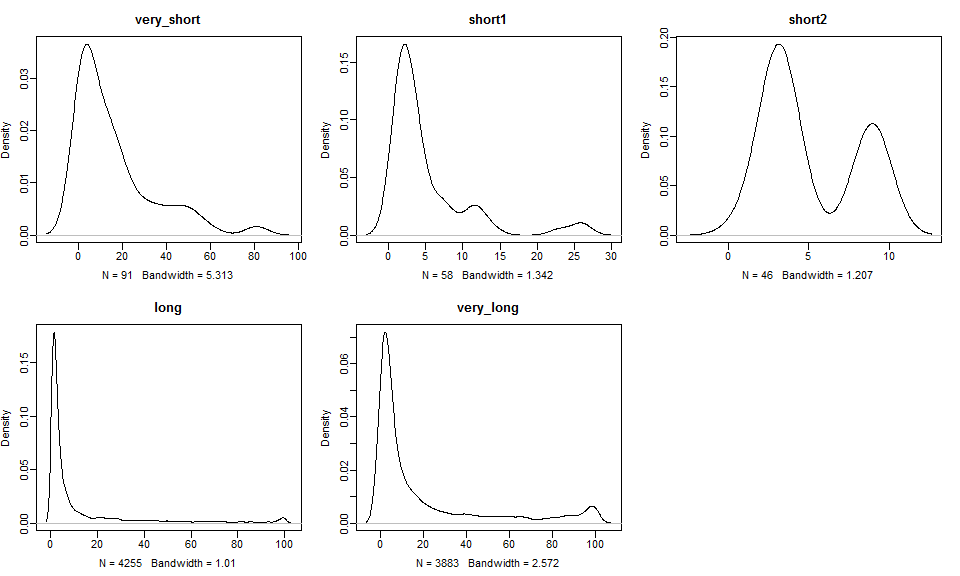
p\_and\_q <- lapply(length\_results, function(x){  
 pvals <- unlist(sapply(x, `[[`, "pval"))  
 qvals <- unlist(sapply(x, `[[`, "adj\_pval"))  
 data.frame(pvals, qvals)  
})  
  
par(mfrow = c(length(p\_and\_q), 2))  
  
  
plot\_density\_highlight <- function(data\_values,   
 xlabel = "",   
 threshold = 0.05,   
 title = "",   
 colour = 3  
 ){  
   
 dens <- density(data\_values)  
 filt <- dens$x < threshold   
   
 plot(dens,   
 main = title,   
 xlab = xlabel,  
 ylim = c(0, max(dens$y) \* 1.2)   
 )  
 polygon(  
 c(dens$x[filt], threshold),   
 c(dens$y[filt], 0),   
 col = colour  
 )  
 text\_label = paste0("n = ", sum(data\_values < threshold))  
 text(dens$x[length(dens$x)/10],   
 y = max(dens$y) \* 1.1,   
 labels = text\_label,   
 font = 2,   
 col = "red2"  
 )  
}  
  
sapply(names(p\_and\_q), function(x) {   
   
 x\_suffix <- paste0("values, N = ", nrow(p\_and\_q[[x]]))  
   
 plot\_density\_highlight(p\_and\_q[[x]]$pvals,   
 title = x,   
 xlabel = paste0("p ", x\_suffix)  
 )  
  
 plot\_density\_highlight(p\_and\_q[[x]]$qvals,   
 title = x,   
 xlabel = paste0("corrected p ", x\_suffix)  
 )  
})



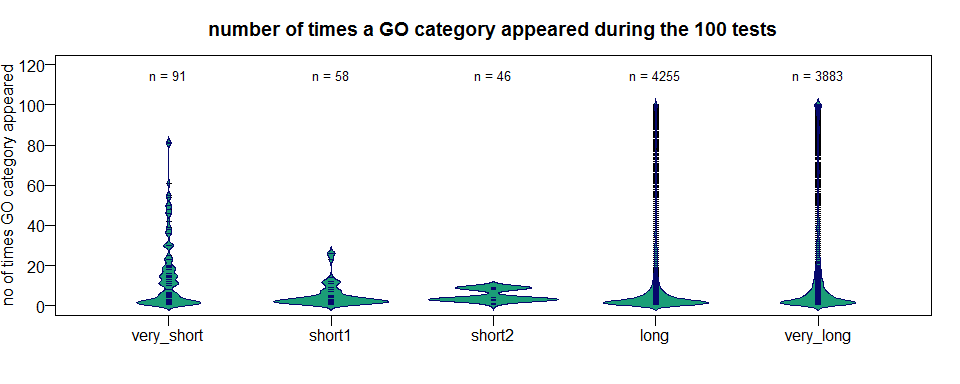
ordered\_categories <- lapply(length\_results, function(length\_subset){  
   
 all\_sig\_categories <- unlist(sapply(length\_subset, rownames))  
 tabled\_categories <- table(all\_sig\_categories)  
 tabled\_categories[order(tabled\_categories, decreasing = TRUE)]  
})  
  
#lapply(ordered\_categories, head, n = 10)

Plot how many times a category appeared.

par(mfrow = c(2, 3))  
sapply(names(ordered\_categories), function(x){  
 plot(density(ordered\_categories[[x]]), main = x)  
})



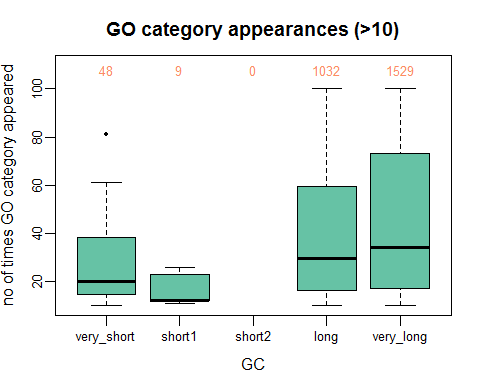
par(mfrow = c(1,1))  
  
beanplot(  
 ordered\_categories,   
 what = c(0,1,0,1),   
 col = c("#1B9E77","#06086d"),   
 ll = 0.03,   
 method = "jitter",   
 border = "#06086d",  
 las = 1,  
 log = "",  
 ylim = c(0, 120),  
 main = "number of times a GO category appeared during the 100 tests",  
 ylab = "no of times GO category appeared"  
)  
  
text(1:length(ordered\_categories),   
 y = 115,   
 cex = 0.8,   
 labels = paste0("n = ", sapply(ordered\_categories, length))  
 )



filtered\_categories <- lapply(ordered\_categories, function(x) x[x >= 10])

# convert list of table to list of vectors  
convert\_tbl\_vec <- function(list\_of\_tables){  
 list\_of\_vec <- lapply(names(list\_of\_tables), function(x){  
   
 vector <- as.vector(list\_of\_tables[[x]])  
 names(vector) <- names(list\_of\_tables[[x]])  
 vector  
 })  
 names(list\_of\_vec) <- names(list\_of\_tables)  
 list\_of\_vec  
}

filtered\_categories <- lapply(ordered\_categories, function(x) x[x >= 10])  
  
filtered\_categories\_vec <- convert\_tbl\_vec(filtered\_categories)   
  
boxplot(filtered\_categories\_vec,  
 main = "GO category appearances (>10)",  
 ylab = "no of times GO category appeared",  
 xlab = "GC",  
 pch = 16,  
 cex = 0.5,  
 col = 1,  
 cex.axis = 0.8,  
 ylim = c(10, 110)  
 )  
  
text(1:22,   
 y = 108,   
 cex = 0.8,   
 col = 2,  
 labels = sapply(filtered\_categories\_vec, length)  
 )



Replot just showing the data for categories that appeared >= 10 times

filtered\_categories <- lapply(ordered\_categories, function(x) x[x >= 10])  
  
beanplot(  
 filtered\_categories,   
 what = c(0,1,0,1),   
 col = c("#1B9E77","#06086d"),   
 ll = 0.03,   
 method = "jitter",   
 border = "#06086d",  
 las = 1,  
 log = "",  
 ylim = c(10, 120),  
 main = "GO category appearances (>10)",  
 ylab = "no of times GO category appeared"  
)  
  
text(1:length(ordered\_categories),   
 y = 115,   
 cex = 0.8,   
 labels = paste0("n = ", sapply(filtered\_categories, length))  
 )

Now we could do with some stats to pick a cutoff for the number of times a category appears. Let’s just select an arbitrary value of 30 for now….

Create a dataset that contains these suspect set of categories

suspects <- lapply(ordered\_categories, function(x) names(x[x >= 10]))   
  
sapply(suspects, length)

## very\_short short1 short2 long very\_long   
## 48 9 0 1032 1529

print\_sums <- function(set1, set2, not\_in = FALSE) {  
 ifelse(  
 not\_in,   
 print(sum(!set1 %in% set2)),   
 print(sum(set1 %in% set2))  
 )  
}  
  
with(suspects, {  
 print\_sums(short1, very\_short)  
 print\_sums(short1, very\_short, TRUE)  
 print\_sums(very\_short, short2)  
 print\_sums(long, very\_long)  
 print\_sums(long, very\_long, TRUE)  
})

## [1] 9  
## [1] 0  
## [1] 0  
## [1] 919  
## [1] 113

## [1] 113

All of the categories in the short1 category are found in the very short category so these can be reduced to one category. The short2 set do not overlap and can just be renamed to “short”

The very\_long and long categories have a lot of overlapping categories so I’ll remove any from the long category that are found in the very\_long category.

suspects$long <- with(suspects, long[!long %in% very\_long])  
suspects$short1 <- with(suspects, short1[!short1 %in% very\_short])  
#suspects$short1 <- NULL  
#names(suspects) <- gsub(names(suspects), pattern = "short2", replacement = "short")  
sapply(suspects, length)

## very\_short short1 short2 long very\_long   
## 48 0 0 113 1529

suspects <- suspects[sapply(suspects, length) > 0]  
sapply(suspects, length)

## very\_short long very\_long   
## 48 113 1529

#, eval=FALSE}  
# write out file with unix line endings  
for (i in 1:length(suspects)){  
   
 ifelse(using\_laptop,  
 filename <- paste0("C:/Users/bigginsl/Desktop/temp/biased\_gene\_lists/identified\_categories/", names(suspects)[i], ".txt"),  
 filename <- paste0("M:/biased\_gene\_lists/identified\_categories/", names(suspects)[i], ".txt")  
 )  
 output\_file <- file(filename, "wb")  
   
 write.table(  
 file = output\_file,  
 x = suspects[[i]],   
 row.names = FALSE,  
 col.names = FALSE,   
 quote = FALSE  
 )  
   
 close(output\_file)  
}