```r
# Determine the frequency of the major allele at every locus and test for significant differences between
genotypes
# in_list: a list of output files from bam-readcount (https://github.com/genome/bam-readcount) converted t
o tables.

prop_test_across_genome <- function(in_list) {

  #Output data frame, containing locus position (pos), minimum and maximum frequency between genotypes (mi
nfreq, maxfreq), p-value for difference (p), and confidence intervals(conf1,2)
  pvals = data.frame(pos=numeric(0), minfreq=numeric(0), maxfreq=numeric(0), p=numeric(0), conf1=numeric(0
), conf2=numeric(0))

  # For every locus
  for (i in 1:nrow(in_list[[1]])) {
    maj_alls = c()
    # Determine major allele for each genotype
    for (j in 1:length(in_list)) {
      A = as.numeric(strsplit(in_list[[j]][i,"A"], ":")[[1]][2])
      C = as.numeric(strsplit(in_list[[j]][i,"C"], ":")[[1]][2])
      G = as.numeric(strsplit(in_list[[j]][i,"G"], ":")[[1]][2])
      t = as.numeric(strsplit(in_list[[j]][i,"T"], ":")[[1]][2])
      max_id = which.max(c(A,C,G,t))
      if (max_id==1) {
        maj_alls[j] = "A"
      } else if (max_id==2) {
        maj_alls[j] = "C"
      } else if (max_id==3) {
        maj_alls[j] = "G"
      } else if (max_id==4) {
        maj_alls[j] = "T"
      }
    }
    # Major allele = the consensus between genotypes
    ma = names(sort(table(maj_alls),decreasing=TRUE)[1])

    maj_all_freqs = c()
    depths = c()

    # Determine major allele freq for each genotype
    for (k in 1:length(in_list)) {
      maj_all_freqs[k] = as.numeric(strsplit(in_list[[k]][i,ma], ":")[[1]][2])
      depths[k] = as.numeric(in_list[[k]][i, "depth"])
    }

    het = which(maj_all_freqs/depths < 0.99)
    maj_all_freqs = maj_all_freqs[het]
    depths = depths[het]

    # Significance testing
    if (length(maj_all_freqs) > 1 ) {
      max_index = which.max(maj_all_freqs/depths)
      min_index = which.min(maj_all_freqs/depths)
      Max = maj_all_freqs[max_index]
      Min = maj_all_freqs[min_index]
      Maxfail = depths[max_index] - Max
      Minfail = depths[min_index]- Min
      prop_matrix = matrix(c(Max, Min, Maxfail, Minfail),ncol=2)
      pvals[i, "minfreq"] = Min/depths[min_index]
      pvals[i, "maxfreq"] = Max/depths[max_index]
      pvals[i, "p"] = fisher.test(prop_matrix, simulate.p.value=TRUE)$p.value
      pvals[i, "conf1"] = fisher.test(prop_matrix, simulate.p.value=TRUE, conf.int=TRUE)$conf.int[1]
      pvals[i, "conf2"] = fisher.test(prop_matrix, simulate.p.value=TRUE, conf.int=TRUE)$conf.int[2]
    }
    pvals[i,"pos"] = in_list[[j]][i,"pos"]
    padj = pvals[complete.cases(pvals),]
    padj$p = p.adjust(padj$p,method="bonferroni")
  }
  return(padj)
}
```