Analysis MHC II DRB

Meinolf Ottensmann

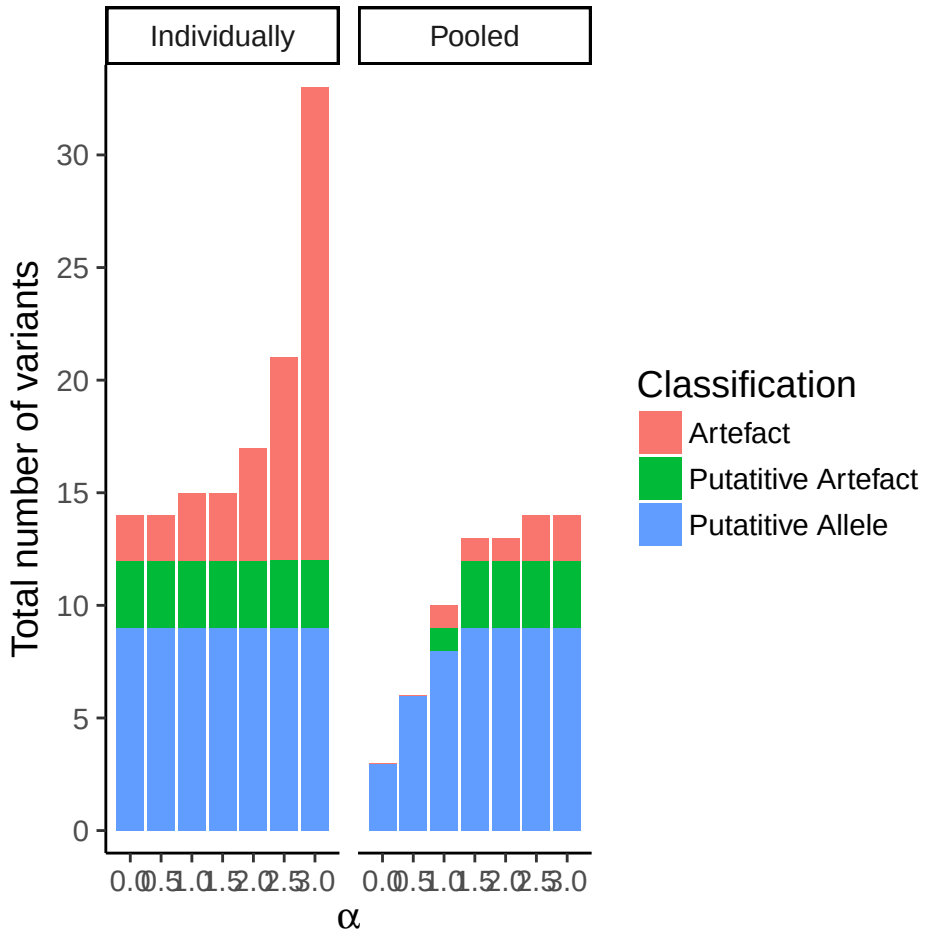
```
library(ggplot2)
library(vegan)
library(magrittr)
library(ape)
library(phangorn)
library(ShortRead)
library(pegas)

source("R/clustering_functions.R")
source("R/fasta_fastq_functions.R")
source("R/genotyping_functions.R")
source("R/summary_stats.R")
```

Explore the influcence of choosen alpha values on clustering results

The impact of the parameter alpha on clustering results was systematically investigated using individual amplicons and pooled amplicons respectively. This anlysis allows to determine how robust allele classification are given the choosen alpha value.

```
## select datasets
fname <-
  c("miseq_reads/DRB-Pool/reference_based/a_0.0_pct_1.0",
    "miseq_reads/DRB-Pool/clustered_reads/alpha_exploration/drb_pct_1.0_a_0.0_ee_1.0",
    "miseq_reads/DRB-Pool/reference_based/a_0.5_pct_1.0",
    "miseq_reads/DRB-Pool/clustered_reads/alpha_exploration/drb_pct_1.0_a_0.5_ee_1.0",
    "miseq_reads/DRB-Pool/reference_based/a_1.0_pct_1.0",
    "miseq_reads/DRB-Pool/clustered_reads/alpha_exploration/drb_pct_1.0_a_1.0_ee_1.0",
    "miseq_reads/DRB-Pool/reference_based/a_1.5_pct_1.0",
    "miseq_reads/DRB-Pool/clustered_reads/alpha_exploration/drb_pct_1.0_a_1.5_ee_1.0",
    "miseq_reads/DRB-Pool/reference_based/a_2.0_pct_1.0",
    "miseq_reads/DRB-Pool/clustered_reads/alpha_exploration/drb_pct_1.0_a_2.0_ee_1.0",
    "miseq_reads/DRB-Pool/reference_based/a_2.5_pct_1.0",
    "miseq_reads/DRB-Pool/clustered_reads/alpha_exploration/drb_pct_1.0_a_2.5_ee_1.0",
    "miseq_reads/DRB-Pool/reference_based/a_3.0_pct_1.0",
    "miseq_reads/DRB-Pool/clustered_reads/alpha_exploration/drb_pct_1.0_a_3.0_ee_1.0")

## genotype in order to classify allele status
out <-
  lapply(fname,
         run_genotyping,
         locus = "drb",
         gain = 0.1,
         doc_min = 45,
         depth_min = 0.7)

names(out) <-
  paste0("Alpha", rep(seq(0.0,3,0.5), each = 2))
```

```r
## push results in data frame
df <- lapply(out, function(x) {
  reshape2::melt(x$zotu_summary, )
  }) %>%
  do.call("rbind",.)

## format df
df$alpha <-
  rep(seq(0.0,3,0.5), each = 6)
df$group <-
  rep(rep(c("Individually", "Pooled"), each = 3), 7)

## make levels look nicer
levels(df$variable)[levels(df$variable) == "putatitive_artefact"] <-
  "Putatitive Artefact"
levels(df$variable)[levels(df$variable) == "putatitive_allele"] <-
  "Putatitive Allele"

# sort factors
df$variable <-
  factor(df$variable, levels = c("Artefact", "Putatitive Artefact", "Putatitive Allele"))

## make plot
plot1_alpha <-
ggplot(df, aes(x = alpha, y = value, fill = variable)) +
  geom_col() +
  theme_classic(base_size = 14) +
  facet_grid(~group) +
  ylab("Total number of variants") +
  xlab(expression(alpha)) +
  guides(fill = guide_legend(title = "Classification")) +
  scale_y_continuous(expand = c(0,1),
                     breaks = seq(0,50,5)) +
  scale_x_continuous(expand = c(0.05,0),
                     breaks = seq(0,3,0.5))
plot1_alpha
```



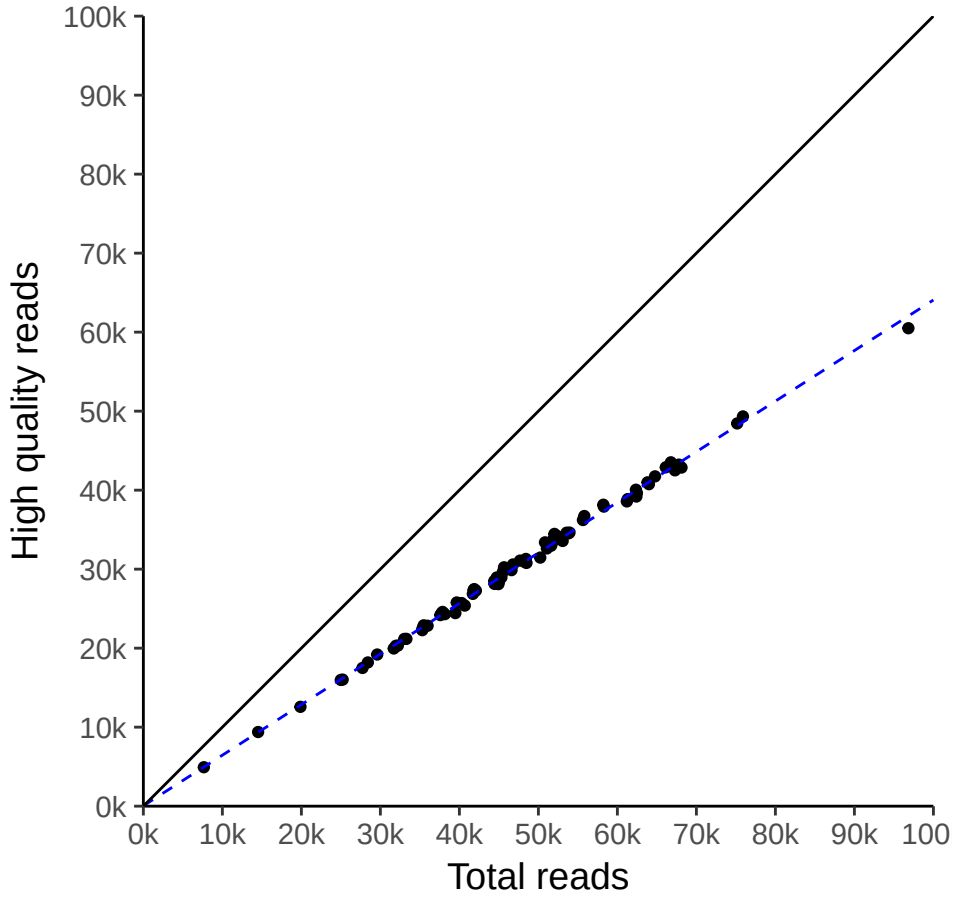
Read and process clustering results

```
## choose data obtained from pooled sequences at alpha = 2.0
fname <- "miseq_reads/DRB-Pool/clustered_reads/alpha_exploration/drb_pct_1.0_a_2.0_ee_1.0"
drb_data <- process_otus(fname, locus = 'drb')
```

Correlation between reads counts

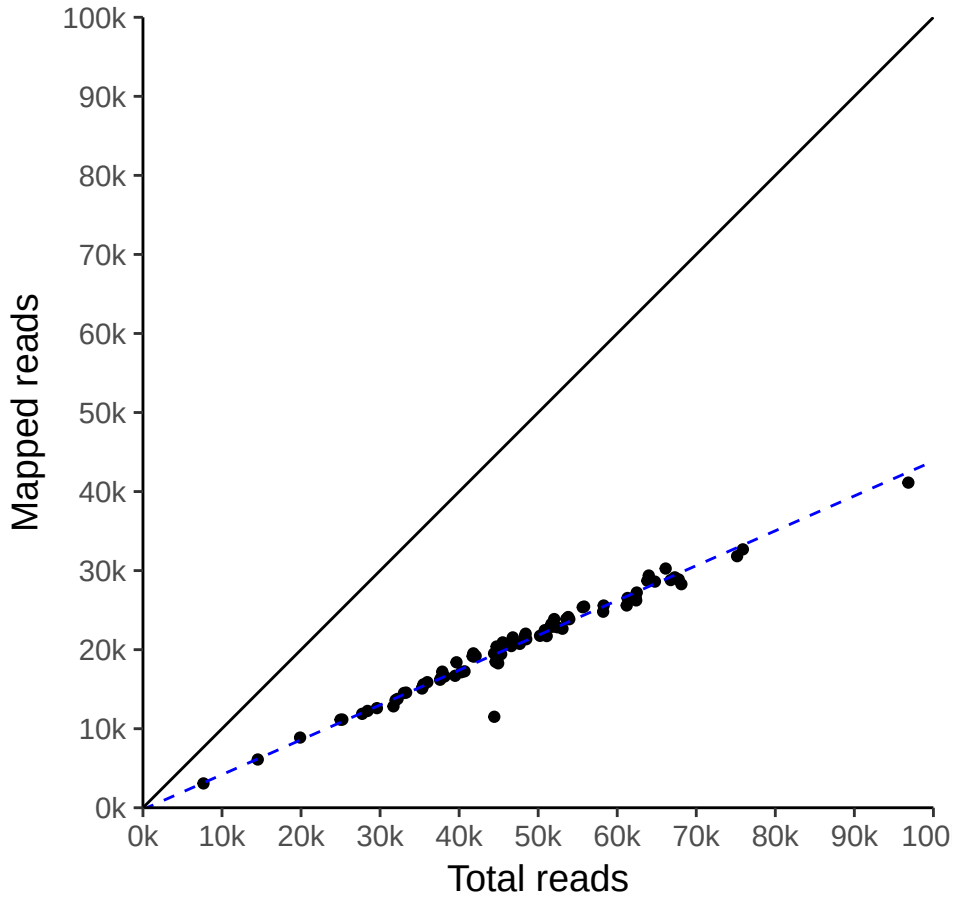
```
## relate filtered read depth to total sequence number
lm_fit <- with(drb_data$bc_counts, lm(Filtered_total~Raw_total))

## correlation
with(drb_data$bc_counts, cor.test(Filtered_total, Raw_total))
>
>   Pearson's product-moment correlation
>
> data:  Filtered_total and Raw_total
> t = 184.04, df = 79, p-value < 2.2e-16
> alternative hypothesis: true correlation is not equal to 0
> 95 percent confidence interval:
```

```
>  0.9981861 0.9992530
> sample estimates:
>       cor
> 0.9988359

## check barcode quality among amplicons
plot2_barcode_quality <-
ggplot(drb_data$bc_counts, aes(x = Raw_total/1000, y = Filtered_total/1000)) +
  geom_point() +
  geom_abline(intercept = lm_fit$coefficients[[1]]/1000,
              slope = lm_fit$coefficients[[2]],
              linetype = "dashed",
              col = "blue") +
  geom_abline(intercept = 0,
              slope = 1) +
  theme_classic(base_size = 14) +
  theme(plot.title = element_text(hjust = 0.5),
        aspect.ratio = 1) +
  xlab("Total reads") +
  ylab("High quality reads") +
  scale_x_continuous(
    expand = c(0,0),
    limits = c(0, floor(max(drb_data$bc_counts$Raw_total)/1000)),
    breaks = seq(0,floor(max(drb_data$bc_counts$Raw_total)/1000),10),
    labels = paste0(seq(0,floor(max(drb_data$bc_counts$Raw_total)/1000),10),"k"))  +
  scale_y_continuous(
    expand = c(0,0),
    limits = c(0, floor(max(drb_data$bc_counts$Raw_total)/1000)),
    breaks = seq(0,floor(max(drb_data$bc_counts$Raw_total)/1000),10),
    labels = paste0(seq(0,floor(max(drb_data$bc_counts$Raw_total)/1000),10),"k"))
plot2_barcode_quality
```




```
## relate mapped read depth to total sequence number
lm_fit <- with(drb_data$bc_counts, lm(mapped~Raw_total))

## correlation
with(drb_data$bc_counts, cor.test(mapped, Raw_total))
>
>   Pearson's product-moment correlation
>
> data:  mapped and Raw_total
> t = 53.348, df = 79, p-value < 2.2e-16
> alternative hypothesis: true correlation is not equal to 0
> 95 percent confidence interval:
>  0.9788870 0.9912554
> sample estimates:
>       cor
> 0.9864032

plot3_barcode_mapping <-
ggplot(drb_data$bc_counts, aes(x = Raw_total/1000, y = mapped/1000)) +
  geom_point() +
  geom_abline(intercept = lm_fit$coefficients[[1]]/1000,
              slope = lm_fit$coefficients[[2]],
              linetype = "dashed",
```

```
              col = "blue") +
  geom_abline(intercept = 0,
              slope = 1) +
  theme_classic(base_size = 14) +
  theme(plot.title = element_text(hjust = 0.5),
        aspect.ratio = 1) +
  xlab("Total reads") +
  ylab("Mapped reads") +
  scale_x_continuous(
    expand = c(0,0),
    limits = c(0, floor(max(drb_data$bc_counts$Raw_total)/1000)),
    breaks = seq(0,floor(max(drb_data$bc_counts$Raw_total)/1000),10),
    labels = paste0(seq(0,floor(max(drb_data$bc_counts$Raw_total)/1000),10),"k")) +
  scale_y_continuous(
    expand = c(0,0),
    limits = c(0, floor(max(drb_data$bc_counts$Raw_total)/1000)),
    breaks = seq(0,floor(max(drb_data$bc_counts$Raw_total)/1000),10),
    labels = paste0(seq(0,floor(max(drb_data$bc_counts$Raw_total)/1000),10),"k"))
plot3_barcode_mapping
```



Assing genotypes to individuals

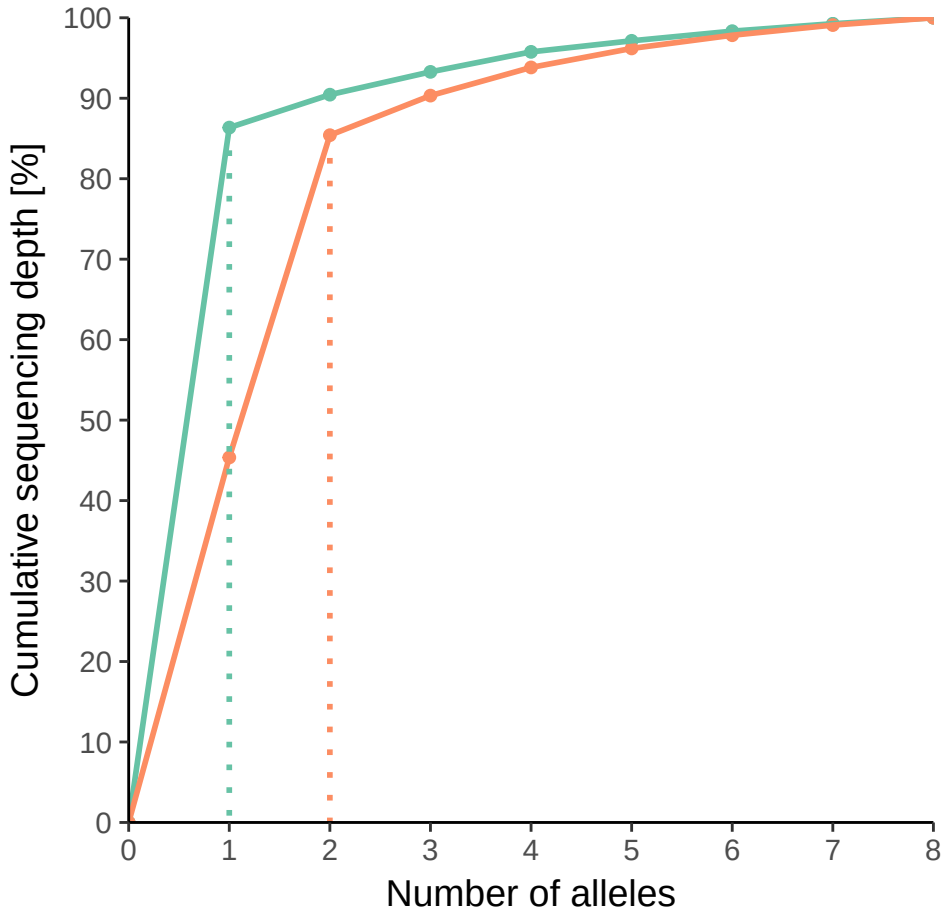
For almost all individuals sequences are assigned to each of the clustered alleles. This is expected for the followign reasons:

- Cross-talk/ tag switching during sequencing
- Spurios reads caused by sequencing and PCR error
- Cross-amplicon contamination

For these reasons, genotyping requires to separate true alleles from artefacts. True alleles are expected to be more common than any of the spurious reads present in a given amplicon. This motivates to use the degree of change (DOC) approach suggested by (2014) that determines inflection points in the cumulative sequencing depth. In contast to commonly used methods that are based on the sequencing depth (Babik et al. 2009, Galan et al. (2010)), this approach does not rely on any arbitrary cut-off value but directly tests the main genotyping assumption outlined above.

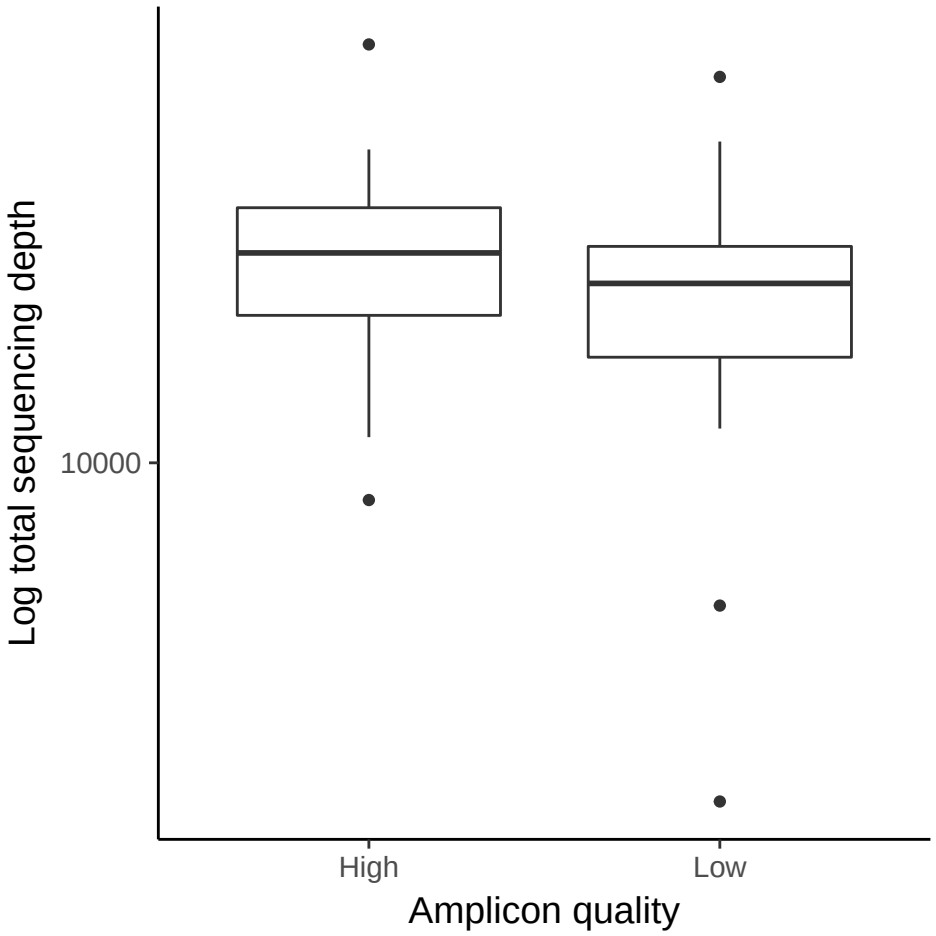
```
## Calculate cumulative sequencing depth for every amplicon.
## Then, call alleles and assign quality class.
genotypes_list <- apply(drb_data$otu_tab, 2,
                        get_genotypes,
                        names = rownames(drb_data$otu_tab),
                        gain = 0.1,
                        doc_min = 40,
                        depth_min = 0.7)
## get cumulative sums
genotypes_df <- do.call("rbind", lapply(genotypes_list, function(x) x[["coord"]])) %>%
  subset(., quality == "High")
genotypes_df$group <- as.factor(genotypes_df$group)

summary(genotypes_df$group)
>   1   2
>  32 392

## Calculate mean relative cumulative sequencing depth
df <- summary_stats(data = genotypes_df, measurevar = "y", groupvars = c("x", "group"))

# add point x = 0, y = 0 for visualisation
df_head <- matrix(0,
                  nrow = length(levels(df$group)),
                  ncol = ncol(df)) %>%
  as.data.frame() %>%
  set_colnames(., colnames(df))
df_head$group <- 1:nrow(df_head)
df <- rbind(df_head, df)

## Get mean depth for all allele number configurations
df_dashes <- data.frame(x = 1:2,
                        group = as.character(1:2),
                        y = c(df$y[df$group == 1 & df$x == 1],
                              df$y[df$group == 2 & df$x == 2]))

plot4_cumul_depth <- ggplot(df, aes(x = x, y = y, col = group)) +
  geom_point(size = 1.75) +
  geom_line(size = 1) +
  geom_segment(data = df_dashes,
```

```
              aes(xend = x, yend = 0),
              linetype = "dotted",
              size = 1) +
  theme_classic(base_size = 14) +
  theme(aspect.ratio = 1,
        legend.position = "none") +
  xlab("Number of alleles") +
  ylab("Cumulative sequencing depth [%]") +
  scale_y_continuous(expand = c(0,0),
                     breaks = seq(0,100,10)) +
  scale_x_continuous(expand = c(0,0),
                     breaks = 0:8) +
  scale_color_brewer(palette = "Set2")
plot4_cumul_depth
```



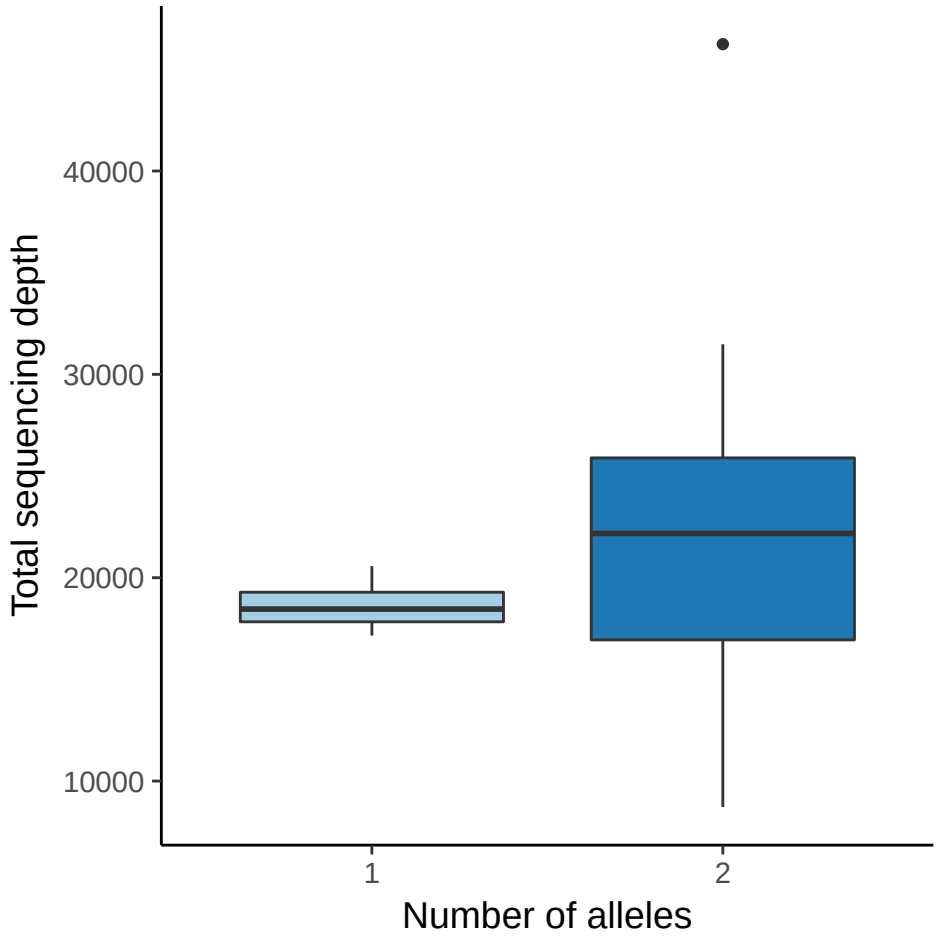

```
## export plots for each sample
  genotypes <- apply(drb_data$otu_tab, 2,
                     get_genotypes,
                     names = rownames(drb_data$otu_tab),
                     gain = 0.1,
                     doc_min = 45,
                     depth_min = 0.7,
```

```
                      plot = T)
  for (i in 1:length(genotypes)) {
    genotypes[[i]] <- genotypes[[i]] + ggtitle(names(genotypes)[i])
  }

pdf("miseq_reads/DRB-Pool/figures/genotypes_DOC.pdf")
for (i in  1:length(genotypes)) print(genotypes[[i]])
dev.off()
> pdf
>   2

## Summarise by amplicon sequencing depth
sequencing_depth <- do.call("rbind", lapply(genotypes_list, function(x) x[["df"]])) %>%
  as.data.frame()

## amplicon quality scores
summary(sequencing_depth$quality)
> High  Low
>   53   28

plot5_amplicon_quality <- ggplot(sequencing_depth, aes(x = quality, y = total_depth)) +
  geom_boxplot() +
  theme_classic(base_size = 14) +
  xlab("Amplicon quality") +
  ylab("Log total sequencing depth") +
  scale_y_log10()
plot5_amplicon_quality
```




```
## no difference
with(sequencing_depth, wilcox.test(total_depth~quality))
>
>   Wilcoxon rank sum test with continuity correction
>
> data:  total_depth by quality
> W = 910, p-value = 0.09624
> alternative hypothesis: true location shift is not equal to 0

allele_num_df <- lapply(genotypes_list, function(x) x[["df"]]) %>%
  do.call("rbind",.) %>%
  subset(., quality == "High")
allele_num_df$row <- rownames(allele_num_df)

plot6_allele_num_depth <-
  ggplot(allele_num_df,
         aes(x = as.factor(n_alleles), y = total_depth,
            fill = as.factor(n_alleles))) +
  geom_boxplot() +
  scale_fill_brewer(palette = "Paired") +
  theme_classic(base_size = 14) +
  theme(legend.position = "none") +
  xlab("Number of alleles") +
```

```
  ylab("Total sequencing depth")
plot6_allele_num_depth
```




```
## no difference
with(allele_num_df, wilcox.test(total_depth~n_alleles))
>
>   Wilcoxon rank sum test
>
> data:  total_depth by n_alleles
> W = 62, p-value = 0.2423
> alternative hypothesis: true location shift is not equal to 0
```

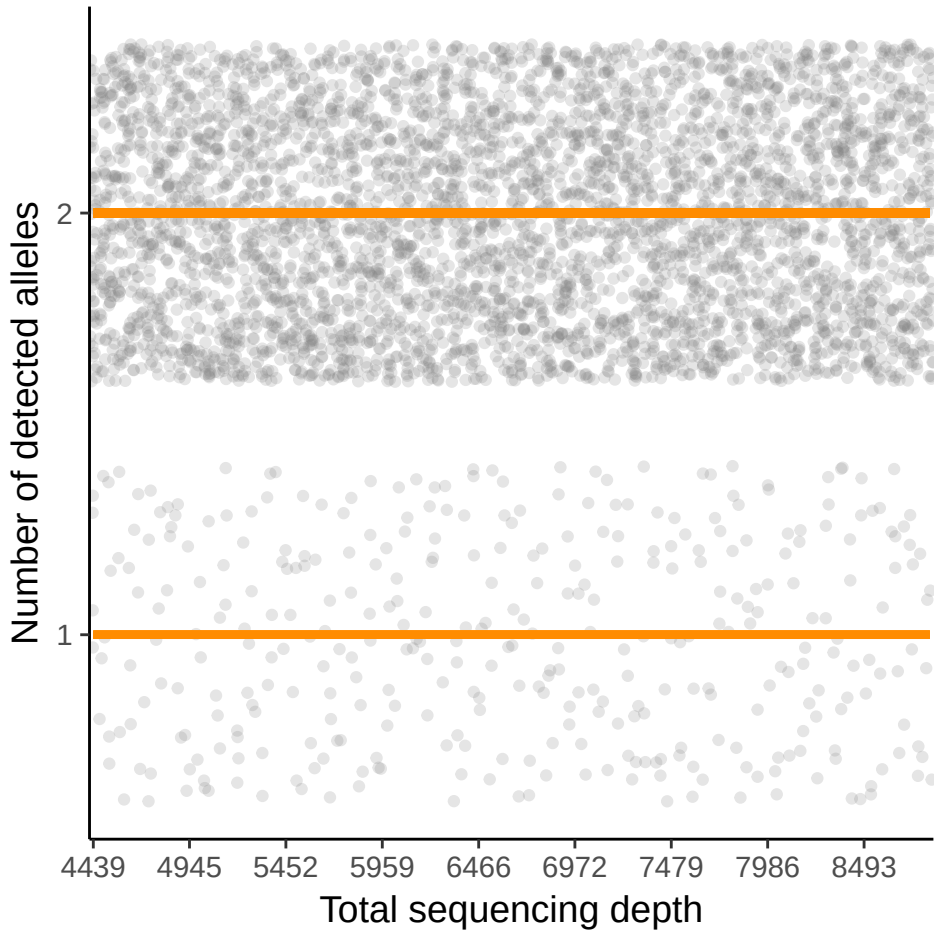
The above boxplot shows that there is no signifcant linear trend of increasing number of allels with respect to the total sequencing depth.

Exploring potential bias by variation in sequencing depth

The function `vegan::rarefy` gives the expected species richness in random subsamples of size sample and therefore allows to test if variation in sequencing depth could have an effect on the detection of alleles.

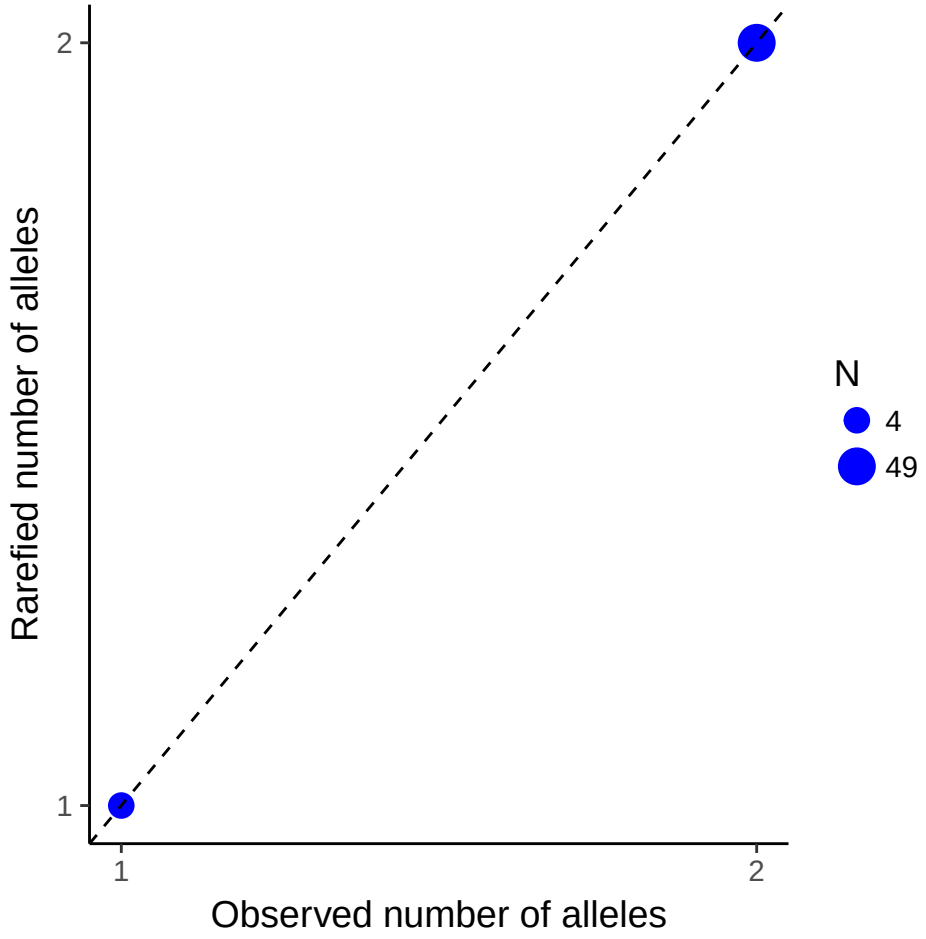
```
## get sequence counts of high quality amplicons
otu_table <-
```

```
  drb_data$otu_tab[rownames(allele_num_df[allele_num_df$quality == "High",])] %>%
  t()

## define sample sizes. Consider 0.5 of minimum depth as starting point
size <- seq(from = floor(min(rowSums(otu_table))*0.5),
                to = min(rowSums(otu_table)),
                by = 50)

## Conduct rarefaction analysis for each sample size
df <- data.frame(
  sample = rep(rownames(otu_table), length(size)),
  size = rep(size, each = length(rownames(otu_table))),
  y = unlist(lapply(size, function(x) {
    rarefaction(m = otu_table,
                n = x,
                gain = 0.1,
                doc_min = 45,
                depth_min = 0.7)})))

plot7_rarefy <- ggplot(df, aes(x = size, y = y, grp = sample)) +
  geom_jitter(col = "grey50", alpha = 0.2) +
  geom_line(size = 1.5, col = "darkorange") +
  theme_classic(base_size = 14) +
  scale_x_continuous(
    expand = c(0,0),
    breaks = floor(seq(from = min(size),
                 to = plyr::round_any(max(size), 1000, f = ceiling),
                 length.out = 10))) +
  scale_y_continuous(breaks = seq(0, max(df$y))) +
  xlab("Total sequencing depth") +
  ylab("Number of detected alleles")
plot7_rarefy
```




```
## Call alleles for the final dataset
called_alleles <- apply(otu_table, 1, function(x) {
  out <- get_genotypes(x,
                       gain = 0.1,
                       doc_min = 45,
                       depth_min = 0.7)
  out[["alleles"]]
})
save(called_alleles, file = "miseq_reads/DRB-Pool/RData/called_alleles.RData")

## Rarefy to minimum depth
df <- data.frame(
  obs = unlist(lapply(called_alleles, length)),
  rarefied = rarefaction(m = otu_table,
                         n = min(rowSums(otu_table)),
                         gain = 0.1,
                         doc_min = 45,
                         depth_min = 0.7))
## summarise
df_sum <- summary_stats(df, measurevar = "obs", groupvars = c("rarefied"))

plot8_rarefied_depth <-
ggplot(df_sum, aes(x = obs, y = rarefied, size = N)) +
```

```
  geom_point(col = "blue") +
  scale_size(range = c(4, 6),
             limits = range(df_sum$N),
             breaks = unique(df_sum$N),
             name = "N") +
  geom_abline(intercept = 0, slope = 1,
              linetype = "dashed") +
  theme_classic(base_size = 14) +
  xlab("Observed number of alleles") +
  ylab("Rarefied number of alleles") +
  scale_x_continuous(breaks = seq(0,10,1)) +
  scale_y_continuous(breaks = seq(0,10,1))
plot8_rarefied_depth
```

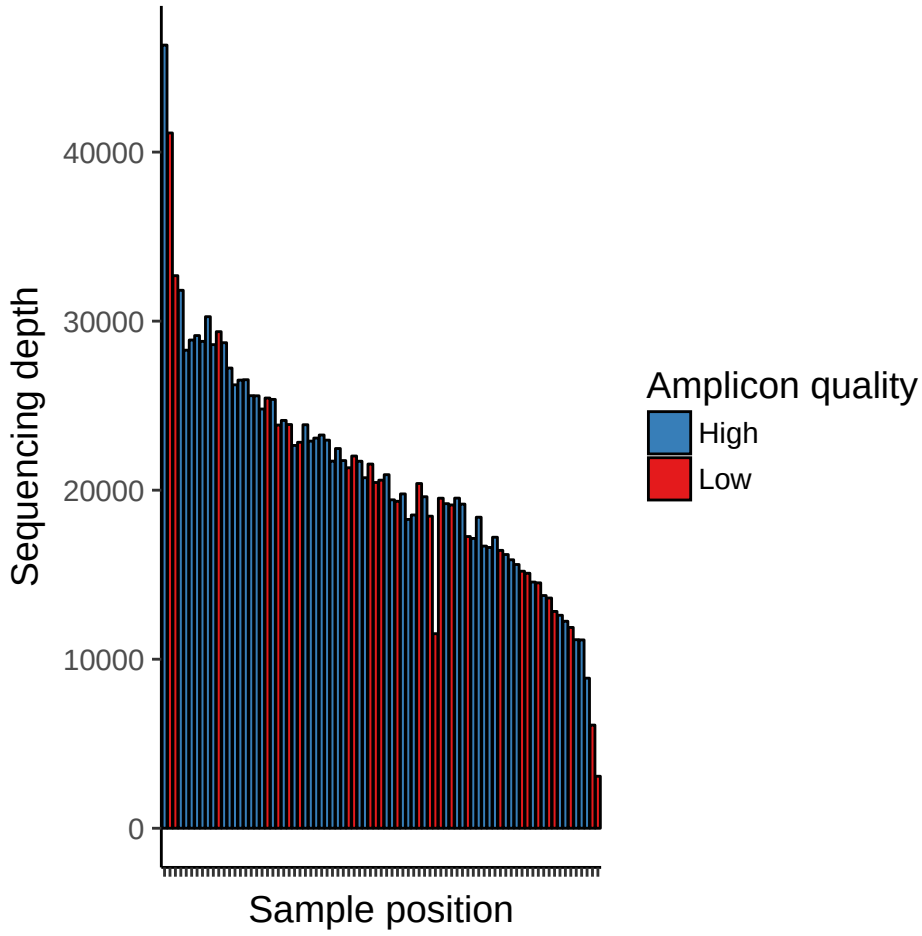


The above graphs show that for all retained samples with a minimum sequencing depth of 8879 genotypes are obustly assigned.

Show Barplot of amplicon sequencing depth and assigned quality

```
df <- drb_data$bc_counts[,c("pos", "mapped")]

df$grp <- ifelse(df$pos %in% names(called_alleles), 'High', 'Low')
```

```
df$pos <- factor(df$pos, levels =  as.character(df$pos))

plot9_retained_samples <- ggplot(df, aes(x = pos, y = mapped, fill = grp)) +
  geom_bar(stat = "identity", colour = "black") +
  theme_classic(base_size = 14) +
  xlab("Sample position") +
  ylab("Sequencing depth") +
  scale_fill_manual(values = c("#377EB8", "#E41A1C")) +
  theme(axis.text.x =  element_blank()) +
  guides(fill = guide_legend(title = "Amplicon quality"))
plot9_retained_samples
```

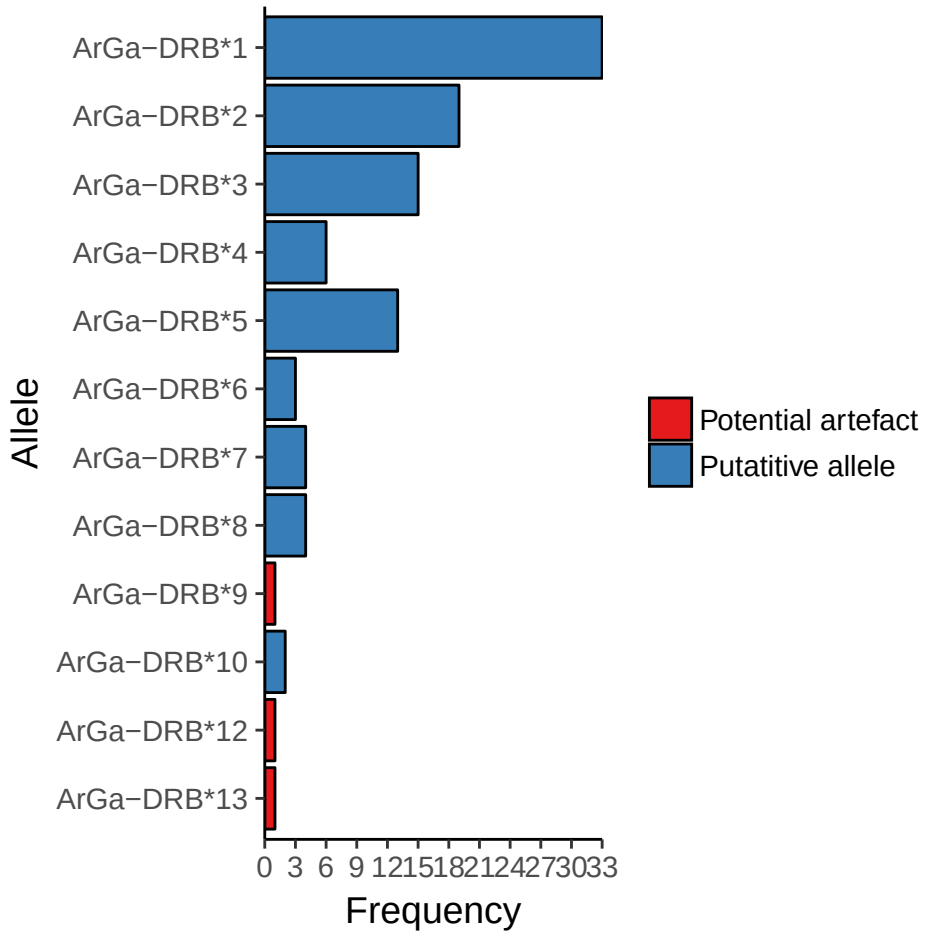



```
summary(as.factor(df$grp))
> High  Low
>   53   28
```

Distribution of alleles

```
df <- unlist(called_alleles) %>%
  as.factor() %>%
  summary()
```

```r
allele_order <- lapply(names(df), function(x) strsplit(x, split = "Zotu")[[1]][2]) %>%
  unlist()  %>%
  as.numeric() %>%
  order(., decreasing = T)

df <- data.frame(x = names(df), y = df)
df$x <- unlist(lapply(df$x, function(x) {
  stringr::str_replace(x, "Zotu", "ArGa-DRB*")
  }))
df$x <- factor(df$x, levels = df$x[allele_order])
df$z <- ifelse(df$y == 1,  "Potential artefact", "Putatitive allele")


plot10_allele_freq <- ggplot(df, aes(x = x, y = y, fill = z)) +
  geom_bar(stat = "identity", colour = "black") +
  theme_classic(base_size = 14) +
  xlab("Allele") +
  ylab("Frequency") +
  scale_y_continuous(expand = c(0,0),
                     breaks = seq(0, max(df$y), by = 3)) +
  scale_fill_brewer(palette = "Set1") +
  guides(fill = guide_legend(title = "")) +
  coord_flip()
plot10_allele_freq
```



Replace rack positions by id names

```
## get mapping file
load("miseq_reads/DRB-Pool/RData/drb_mapping_file.RData")
drb_mapping_file <-
  subset(drb_mapping_file, as.character(pos) %in% names(called_alleles))
## get factors
factors <-
  read.table(file = "data/factors.txt") %>%
  subset(.,rownames(.) %in% drb_mapping_file$Pair)
## set pair id as rowname
factors$Pair <- rownames(factors)

## merge data sheets
drb_mapping_file <- dplyr::left_join(drb_mapping_file, factors, by = "Pair")
drb_mapping_file$Pair <- paste0(drb_mapping_file$Pair,"_" , 1:nrow(drb_mapping_file))

## substitute rack location by pair sample identifier
called_alleles_renamed <- called_alleles
names(called_alleles_renamed) <-
  drb_mapping_file$Pair[match(names(called_alleles), drb_mapping_file$pos)]
```

```
## get sample names
samples <- lapply(names(called_alleles_renamed), function(x) strsplit(x, "_")[[1]][1])

## find replicated samples
replicated <- samples[duplicated(samples)]

## For samples P33 & P24 the Replicates D11 & E11 were swapped. When corrected,
## the genotypes are 100 % repeatable and mums-pups share 0.5
## replicates are removed for downstram analyses

remove <- which(names(called_alleles) %in% c("D11", "E11"))
genotypes_drb <- called_alleles_renamed[-remove]

## remove name extensions
names(genotypes_drb) <-
  lapply(names(genotypes_drb), function(x) strsplit(x, "_")[[1]][1]) %>%
  unlist()

save(genotypes_drb, file = "miseq_reads/DRB-Pool/RData/genotypes_drb.RData")
```

Median Joining network

```
x <- fasta2mat(
  fasta = "miseq_reads/DRB-Pool/clustered_reads/drb_pct_1.0_a_2.0_ee_1.0.fixed.otus.fa")

# Median-Joining Network
mjn1 <- mjn(x = x, prefix = "", epsilon = 0)

pdf(file = "miseq_reads/DRB-Pool/plots/mjn.pdf",
    width = 14,
    height = 14)
plot(mjn1, col.link = "grey80", pie = matrix(1, nrow = length(attr(mjn1, "labels"))),
     font = 1, cex = 0.8, scale.ratio = 1)
dev.off()
```

Mapping alleles to Genome and Transcriptome

Alleles map all to Contig 48. The top hit represents the expected location of the DRB locus, whereas the third hit shows the DQB locus.

```
blastn
  -db linux/db/arc_gaz_genome_db
  -outfmt 6
  -evalue 1e-8
  -word_size 7
  -query miseq_reads/DRB-Pool/clustered_reads/drb_pct_1.0_a_2.0_ee_1.0.fixed.otus.fa
  -out  miseq_reads/DRB-Pool/fasta/drb2_arc_gaz_genome.fasta

blastn
  -db linux/db/arc_gaz_transcriptome_db
  -outfmt 6
```

```
  -evalue 1e-8
  -word_size 7
  -query miseq_reads/DRB-Pool/clustered_reads/drb_pct_1.0_a_2.0_ee_1.0.fixed.otus.fa
  -out  miseq_reads/DRB-Pool/fasta/drb2_arc_gaz_transcriptome.fasta
```

```
read.table("miseq_reads/DRB-Pool/fasta/drb2_arc_gaz_genome.fasta")[,c(1:4,9,10)] %>%
  set_colnames(., value = c("Allele", "Contig","Identity","Alignment", "Start", "End")) %>%
  head()
>   Allele   Contig Identity Alignment   Start     End
> 1  Zotu1 Contig48    94.00       200 2002565 2002761
> 2  Zotu1 Contig48    90.40       198 1842516 1842325
> 3  Zotu1 Contig48    90.80       163 1937299 1937137
> 4  Zotu2 Contig48    92.75       193 2002565 2002754
> 5  Zotu2 Contig48    92.23       193 1842516 1842330
> 6  Zotu2 Contig48    90.18       163 1937299 1937137
```

Alleles map to a single region of the assembled transcriptome that is identical to the DQB top hit.

```
read.table("miseq_reads/DRB-Pool/fasta/drb2_arc_gaz_transcriptome.fasta")[,c(1:4,9,10)] %>%
  set_colnames(., value = c("Allele", "Contig","Identity","Alignment", "Start", "End")) %>%
  head()
>   Allele         Contig Identity Alignment Start End
> 1  Zotu1 AgU032193_v1.1    91.41       163   208 370
> 2  Zotu2 AgU032193_v1.1    94.48       163   208 370
> 3  Zotu3 AgU032193_v1.1    93.25       163   208 370
> 4  Zotu4 AgU032193_v1.1    91.41       163   208 370
> 5  Zotu5 AgU032193_v1.1    92.64       163   208 370
> 6  Zotu6 AgU032193_v1.1    93.87       163   208 370
```

Check transcriptome reads for expression of alleles

In order to check for evidence of gene expression for the newly characterised, alleles are mapped to raw transcriptome reads published by (Hoffman et al. 2013). These sequences are available as an archieve on Genbank under accession number SRA064103. Reads are available as separate files that will be firt downloaded and then merge into a single file that is ready to use for blasting. Here the maximum number of 45 alleles retained after clustering amplicons individually with (alpha = 3.0) all 45 alleles that include 18 putative alleles, 3 putatitive artefacts and 24 sequences without support.

```
## bash
cd DQB

## Download 454 raw reads
## ----------------------
vdb-dump -f tab -C READ SRR646623 | awk '{print ">" "heart." NR "\n" $0}'
         > heart.SRA064103.fasta &
vdb-dump -f tab -C READ SRR646624 | awk '{print ">" "intestine." NR "\n" $0}'
         > intestine.SRA064103.fasta &
vdb-dump -f tab -C READ SRR646625 | awk '{print ">" "kidney." NR "\n" $0}'
         > kidney.SRA064103.fasta &
vdb-dump -f tab -C READ SRR646626 | awk '{print ">" "lung." NR "\n" $0}'
         > lung.SRA064103.fasta &
vdb-dump -f tab -C READ SRR646627 | awk '{print ">" "spleen." NR "\n" $0}'
         > spleen.SRA064103.fasta &
vdb-dump -f tab -C READ SRR646628 | awk '{print ">" "testis." NR "\n" $0}'
```

```
        > testis.SRA064103.fasta &

## Merge files
## -----------
cat *.SRA064103.fasta > transcriptome_reads.fasta

## Dereplicate alleles:
## --------------------
usearch10.exe -fastx_uniques arga_dqb.fasta -fastaout arga_dqb_uniques.fasta

## Remove size annotation
## ----------------------
usearch10.exe -fastx_strip_annots arga_dqb_uniques.fasta -fastaout arga_dqb_derep.fasta

## make blast database from allele sequences
## -----------------------------------------
makeblastdb -in arga_dqb_derep.fasta -dbtype nucl -out arga_dqb_db

## Blast 454 reads to alleles
## --------------------------
blastn -db arga_dqb_db -outfmt 6 -evalue 1e-8 -word_size 7
       -query transcriptome_reads.fasta -out transcriptome_reads.arga_dqb.txt &
```

```
blastn_output <-
  read.table("DRB/transcriptome_reads.arga_drb.txt")[,1:6] %>%
  set_colnames(., value = c("Query", "Allele", "Similarity",
             "Length", "Mismatches", "Gaps")) %>%
  subset(., Similarity >= 94 & Length %in% 200:205)

seqs <- readFasta("DRB/transcriptome_reads.fasta")
seqs <- seqs[which(id(seqs) %in% blastn_output$Query)]
writeFasta(object = seqs,
           file = "DRB/transcriptome_reads.arga_drb.fasta")
write.table(x = blastn_output,
            file = "DRB/transcriptome_reads.arga_drb.hits.metafile.txt",
            row.names = F)
```

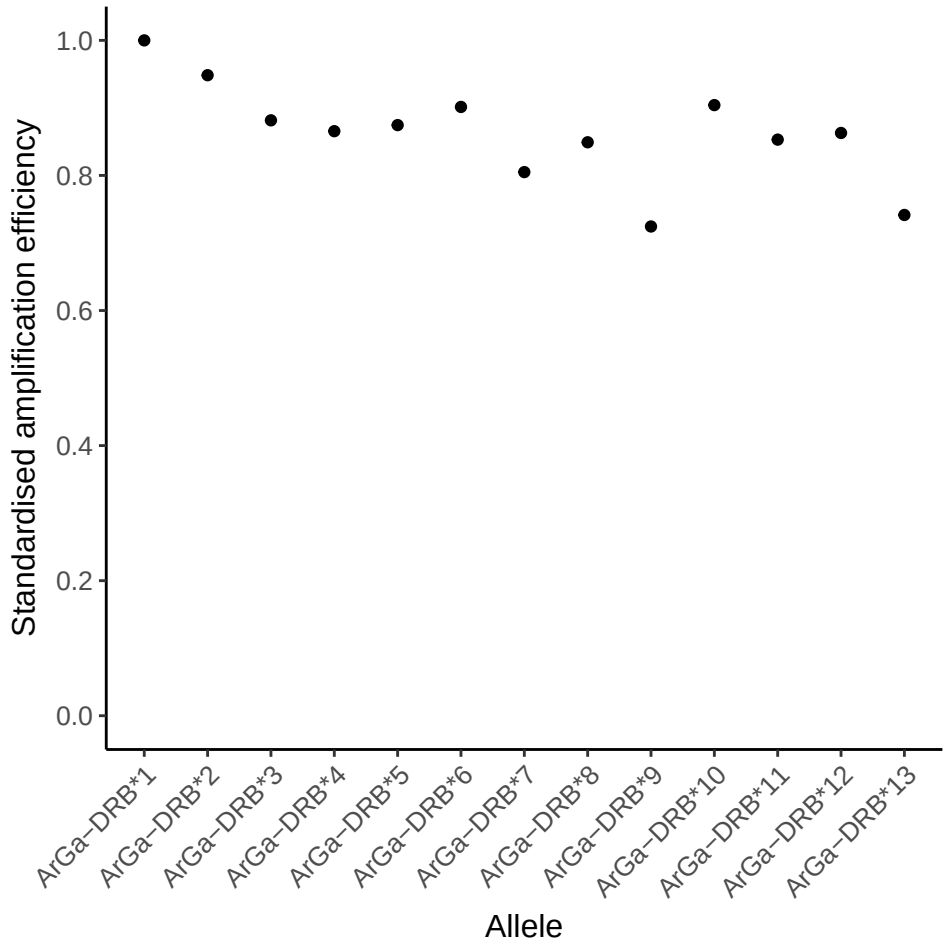
```
expressed <- read.csv("DRB/expressed_arga_drb.csv")
expressed$Allele <- factor(expressed$Allele, levels = paste0("ArGa-DRB*", c(11,5,4,2,1)))

plot22_gene_expression <-
ggplot(expressed, aes(x = Allele, y = 1, fill = Organ)) +
  geom_bar(stat = "identity") +
  theme_classic(base_size = 14) +
  theme(aspect.ratio = 1,
        axis.title = element_text(size = 14),
        axis.text.y = element_text(size = 12),
        axis.text.x =  element_blank(),
        axis.ticks.x = element_blank(),
        axis.line.x = element_blank()) +
  xlab("") +
  ylab("") +
  coord_flip() +
  scale_fill_brewer(palette = "Set2")
```

```
plot22_gene_expression
```

Estimating differential amplification efficiencies across alleles

In a study on MHC II DRB in a rodent, Sommer *et al.* (2013) have shown remarkable variation in the amplfication efficiencies differing by more than a magnitude among allele pairs.

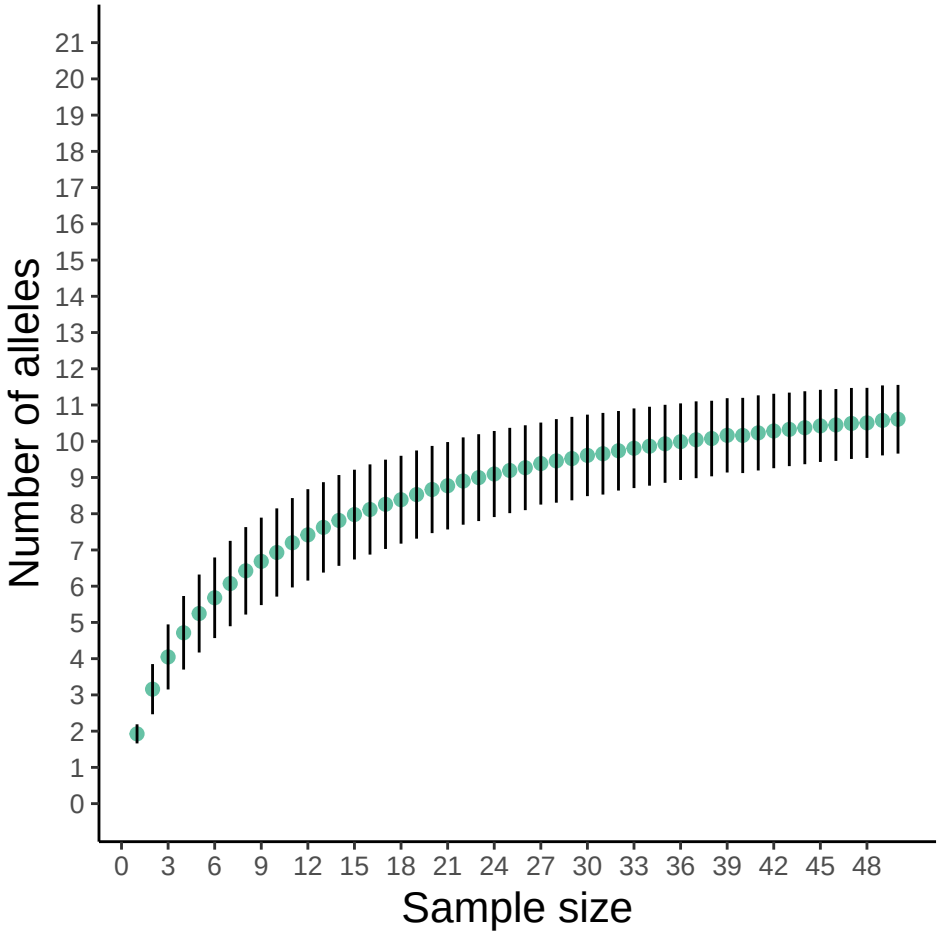
```
otu_table_purified <- purify_otus(x = otu_table,
                                  y = called_alleles)
## number of alleles
nb.alleles <- ncol(otu_table_purified)

## efficiency prior, all are equal i.e. 1
efficiency_prior <- rep(1, nb.alleles)

## Fit relative efficiencies based on Loglikelyhood
efficiency_obs_rel <- optim(par = efficiency_prior,
                            fn = LoglikData,
                            data = otu_table_purified,
                            control = list(fnscale = -1),
                            method = "L-BFGS-B",
                            lower = rep(0.1, nb.alleles),
                            upper = rep(6, nb.alleles))

## Standardise effiencies with respect to ArGa-DRB*1
efficiency_obs_norm <-
  efficiency_obs_rel$par/efficiency_obs_rel$par[which(colnames(otu_table) == "Zotu1")]

## create a data frame
df <- data.frame(allele = colnames(otu_table),
                 efficiency = efficiency_obs_norm)
df$allele <- factor(df$allele, levels = paste0("Zotu", 1:nrow(df)))
levels(df$allele) <- gsub("Zotu", "ArGa-DRB*", levels(df$allele))

plot11_amplification_efficiency <-
ggplot(df, aes(x = allele, y = efficiency)) +
  geom_point() +
  theme_classic() +
  theme(axis.text.x = element_text(angle = 45, vjust = 1, hjust = 1),
        text = element_text(size = 12)) +
  xlab("Allele") +
  ylab("Standardised amplification efficiency") +
  scale_y_continuous(breaks = seq(0,1,0.2),
                     limits = c(0,1))
plot11_amplification_efficiency
```



Allele detection curve

```
sample_alleles <- function(data, n = seq(1, 50,1), bs = 9999) {
  x <- rep(n, each = bs)
  y <- lapply(x, function(temp) {
    get <- data[sample(x = 1:length(data),
             size = temp,
             replace = T)] %>%
  unlist() %>%
  unique() %>%
  length()
  })

  df <- data.frame(x = x,y = unlist(y))
  df$x <- as.factor(df$x)
  return(df)
}

sampled_alleles_df <- sample_alleles(data = called_alleles)
sampled_alleles_df$x <- as.factor(sampled_alleles_df$x)
```

```
df <- summary_stats(sampled_alleles_df,
                    measurevar = "y",
                    groupvars = "x",
                    conf.interval = 0.95)
df$x <- as.numeric(as.character(df$x))

plot12_allele_detection <- ggplot(df, aes(x,y)) +
  geom_point(col = "#66c2a5", size = 2) +
  geom_linerange(ymin = df$y - df$sd, ymax = df$y + df$sd) +
  theme_classic(base_size = 12) +
  theme(aspect.ratio = 1,
        axis.title = element_text(size = 16)) +
  xlab("Sample size") +
  ylab("Number of alleles") +
  scale_x_continuous(breaks = seq(0,50,3)) +
  scale_y_continuous(breaks = seq(0,21,1),
                     limits = c(0,21))
plot12_allele_detection
```




```
> pdf
>   2
```

```
sessionInfo()
> R version 3.4.3 (2017-11-30)
> Platform: x86_64-w64-mingw32/x64 (64-bit)
> Running under: Windows 10 x64 (build 16299)
>
> Matrix products: default
>
> locale:
> [1] LC_COLLATE=English_United Kingdom.1252
> [2] LC_CTYPE=English_United Kingdom.1252
> [3] LC_MONETARY=English_United Kingdom.1252
> [4] LC_NUMERIC=C
> [5] LC_TIME=English_United Kingdom.1252
>
> attached base packages:
> [1] stats4    parallel  stats     graphics  grDevices utils
> [7] datasets  methods   base
>
> other attached packages:
>  [1] pegas_0.10-0.1             adegenet_2.1.0
>  [3] ade4_1.7-8                 ShortRead_1.34.1
>  [5] GenomicAlignments_1.12.2   SummarizedExperiment_1.6.3
>  [7] DelayedArray_0.2.7         matrixStats_0.52.2
>  [9] Biobase_2.36.2             Rsamtools_1.28.0
> [11] GenomicRanges_1.28.4       GenomeInfoDb_1.12.2
> [13] Biostrings_2.44.2          XVector_0.16.0
> [15] IRanges_2.10.2             S4Vectors_0.14.3
> [17] BiocParallel_1.10.1        BiocGenerics_0.22.0
> [19] phangorn_2.3.1             ape_5.0
> [21] magrittr_1.5               vegan_2.4-4
> [23] lattice_0.20-35            permute_0.9-4
> [25] ggplot2_2.2.1              knitr_1.17
>
> loaded via a namespace (and not attached):
>  [1] splines_3.4.3           R.utils_2.6.0
>  [3] gtools_3.5.0            shiny_1.0.5
>  [5] assertthat_0.2.0        expm_0.999-2
>  [7] sp_1.2-5                latticeExtra_0.6-28
>  [9] GenomeInfoDbData_0.99.0 LearnBayes_2.15
> [11] yaml_2.1.16             backports_1.1.2
> [13] glue_1.2.0              quadprog_1.5-5
> [15] digest_0.6.13           RColorBrewer_1.1-2
> [17] colorspace_1.3-2        R.oo_1.21.0
> [19] htmltools_0.3.6         httpuv_1.3.5
> [21] Matrix_1.2-12           plyr_1.8.4
> [23] pkgconfig_2.0.1         gmodels_2.16.2
> [25] zlibbioc_1.22.0         xtable_1.8-2
> [27] scales_0.5.0            gdata_2.18.0
> [29] tibble_1.3.4            mgcv_1.8-22
> [31] lazyeval_0.2.1          mime_0.5
> [33] deldir_0.1-14           evaluate_0.10.1
> [35] R.methodsS3_1.7.1       nlme_3.1-131
> [37] MASS_7.3-47             hwriter_1.3.2
```

```
> [39] tools_3.4.3             stringr_1.2.0
> [41] munsell_0.4.3           cluster_2.0.6
> [43] bindrcpp_0.2            compiler_3.4.3
> [45] rlang_0.1.4             grid_3.4.3
> [47] RCurl_1.95-4.8          igraph_1.1.2
> [49] labeling_0.3            bitops_1.0-6
> [51] rmarkdown_1.8           boot_1.3-20
> [53] gtable_0.2.0            reshape2_1.4.3
> [55] R6_2.2.2                dplyr_0.7.4
> [57] seqinr_3.4-5            bindr_0.1
> [59] fastmatch_1.1-0         rprojroot_1.3-1
> [61] spdep_0.7-4             stringi_1.1.6
> [63] Rcpp_0.12.14            coda_0.19-1
> [65] spData_0.2.6.8
```

References

Babik, Wiesław, Pierre Taberlet, Maciej Jan Ejsmond, and Jacek Radwan. 2009. "New Generation Sequencers as a Tool for Genotyping of Highly Polymorphic Multilocus Mhc System." *Molecular Ecology Resources* 9 (3): 713–19.

Galan, Maxime, Emmanuel Guivier, Gilles Caraux, Nathalie Charbonnel, and Jean-François Cosson. 2010. "A 454 Multiplex Sequencing Method for Rapid and Reliable Genotyping of Highly Polymorphic Genes in Large-Scale Studies." *BMC Genomics* 11 (1): 296.

Hoffman, Joseph I., Michael A. S. Thorne, Philip N. Trathan, and Jaume Forcada. 2013. "Transcriptome of the Dead: Characterisation of Immune Genes and Marker Development from Necropsy Samples in a Free-Ranging Marine Mammal." *BMC Genomics* 14: 52. doi:10.1186/1471-2164-14-52.

Lighten, Jackie, Cock van Oosterhout, and Paul Bentzen. 2014. "Critical Review of Ngs Analyses for de Novo Genotyping Multigene Families." *Molecular Ecology* 23 (16): 3957–72. doi:10.1111/mec.12843.

Sommer, S., A. Courtiol, and C. J. Mazzoni. 2013. "MHC Genotyping of Non-Model Organisms Using Next-Generation Sequencing: A New Methodology to Deal with Artefacts and Allelic Dropout." *BMC Genomics* 14. doi:10.1186/1471-2164-14-542.