SNP filtering using vcfR

D. G. Gannon

September 2020

Previous quality filters up to this point include:

1) Demultiplexing phase: Removing reads for which Phred score falls below 10 in a 15% sliding window (`process_radtags` program)
2) Demultiplexing phase: Removing reads with an uncalled base (`process_radtags` program)
3) Catalog phase: Discard alignments with Phred scores < 20 (i.e. only keep alignments for which Pr(Incorrect alignment < 0.01); `gstacks` program)
4) Genotyping: Loci must be present in at least 85% of individuals per population and all four populations (meadow complexes) sampled
5) Locus filters: vcf file is filtered to include only bi-allelic loci using `vcftools`.

```
vcftools --vcf ./AQFO_snps_r80.vcf --min-alleles 2 --max-alleles 2 --recode --recode-INFO-all \
--out ./AQFO_snps_r80_bi
```

Filtering SNPs based on depth

Repetitive or similar regions in the genome may result in reads that come from different locations in the genome getting erroneously merged and aligned. This can result in exceptionally deep coverage on certain loci and could result in spurious SNP calls. We use the effective per-sample coverage reported from the `gstacks` of `Stacks v2` to create a depth threshold. We set the depth threshold, d^*, as

$$0.0001 = 1 - P(d_{ij} \leq d_i^* \mid \lambda_i),$$

where, λ_i is the reported effective coverage for sample i from the `gstacks` program, d_{ij} is the depth at locus j for sample i, and $P(\cdot)$ is computed based on the Poisson PMF with rate parameter λ_i.

We do not remove SNPs with low depth because the model that calls genotypes in `Stacks v2` avoids calling genotypes when uncertainty in the call is high (Maruki and Lynch 2017).

```
  dp <- extract.gt(col.vcf, element = "DP", as.numeric = T)

# Replace exceedingly high read depths with NAs in depth matrix
  quants <- qpois(0.0001, lambda = coverage$mean_cov_ns, lower.tail = F)
```

```
  dp2 <- sweep(dp, MARGIN = 2, STATS = quants, FUN = "-")

  dp[dp2 > 0] <- NA
```

```
#Filter out the snps with lots of missing data

  miss_snp <- apply(dp, MARGIN = 1, function(x){ sum( is.na(x) ) } )
  miss_snp <- miss_snp / ncol(dp)
  col.vcf.good <- col.vcf[miss_snp <= 0.05, ]



# get a list of snp id's to remove

  site2rm <- col.vcf@fix[miss_snp > 0.05, 3]

 #  write.table(site2rm, "~/Documents/Columbine/Data/GBS_Data/exclude_sites.txt",
 #              quote = F, sep = "\t",
 #            row.names = F, col.names = F)
```

Using the list of SNPs to exclude, we use vcftools to rewrite the vcf file with only the loci we wish to keep.

```
  vcftools --vcf ./AQFO_snps_r80_bi.vcf --exclude ./exclude_sites.txt --recode --recode-INFO-all \
  --out ./AQFO_snps_postfilt
```

```
  dp.good <- extract.gt(col.vcf.good, element = "DP", as.numeric = T)
  col.gt <- extract.gt(col.vcf.good, element = "GT")

  heatmap.bp(dp.good, rlabels = F)
```

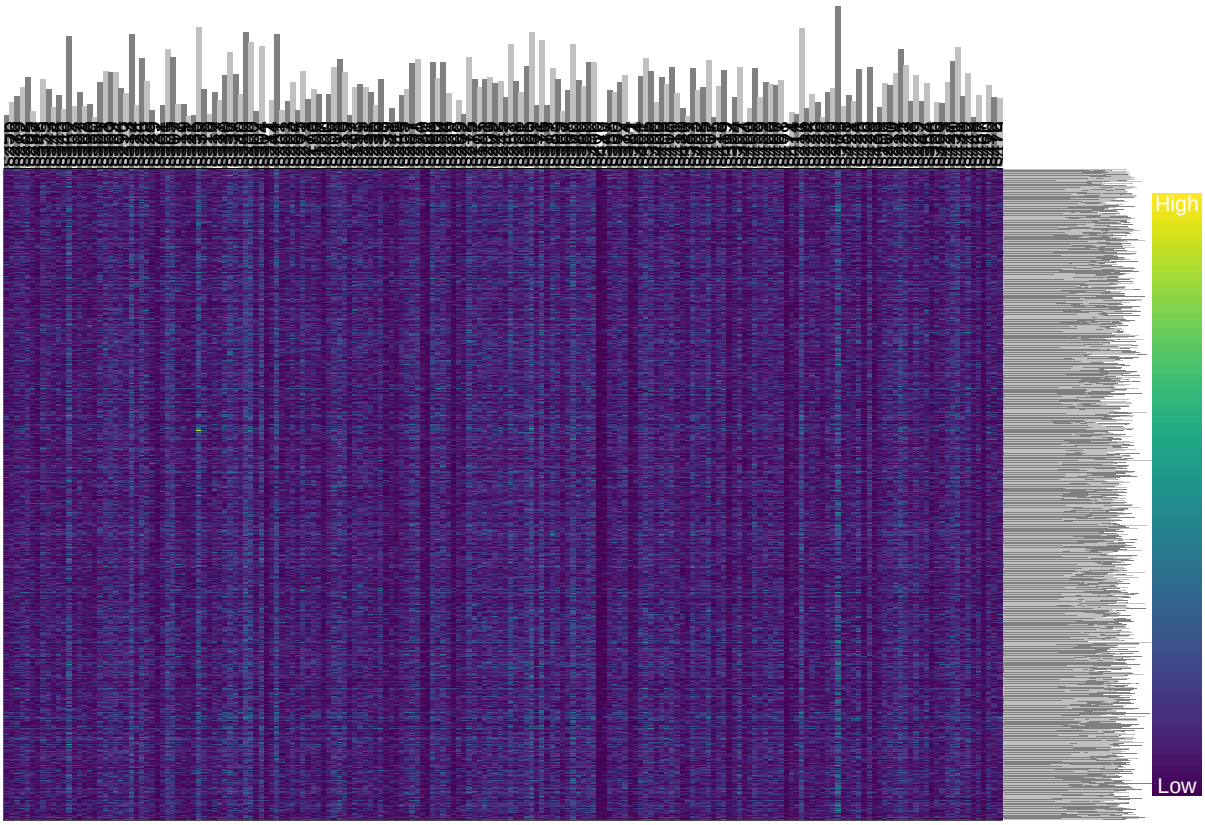

Figure 1 Heatmap of sequencing depth across the filtered dataset.

With this vcf file containing 2.34 percent missing information, 10964 SNPs and 192 individual plants, we impute the missing genotypes using `BEAGLE` (Browning and Browning 2016).

Filtering SNPs based on exon annotations

Read in the annotations file from the *A. coerulea* genome and store as a dataframe.

```
# read in the imputed vcf
  col.vcf.good <- read.vcfR(file = "~/Documents/Columbine/Data/GBS_Data/AQFO_snsp_impute.vcf")
```

```
## Scanning file to determine attributes.
## File attributes:
##   meta lines: 8
##   header_line: 9
##   variant count: 10964
##   column count: 201
## Meta line 8 read in.
## All meta lines processed.
## gt matrix initialized.
## Character matrix gt created.
##   Character matrix gt rows: 10964
```

```
##   Character matrix gt cols: 201
##   skip: 0
##   nrows: 10964
##   row_num: 0
## Processed variant 1000Processed variant 2000Processed variant 3000Processed variant 4000Processed va
## All variants processed
```

```
  ann <- read.table("~/Documents/Columbine/Data/GBS_Data/Acoerulea_322_v3.1.gene_exons.gff3",
                    header = F, sep = "\t", as.is = T)[,c(1,3:5,7)]

# name columns
  names(ann) <- c("Chrom", "annot", "start", "stop", "strand")

# structure of the data
  head(ann)
```

```
##    Chrom          annot start stop strand
## 1 Chr_01           gene  2657 4987      +
## 2 Chr_01           mRNA  2657 4987      +
## 3 Chr_01           exon  2657 2841      +
## 4 Chr_01 five_prime_UTR  2657 2841      +
## 5 Chr_01           exon  4435 4987      +
## 6 Chr_01 five_prime_UTR  4435 4439      +
```

```
# annotate snps
  snps <- as.data.frame(col.vcf.good@fix)
  snps$POS <- as.numeric(snps$POS)
  snps$strand <- str_split(snps$ID, ":", simplify = T)[,3]
  snps$SOFA <- NA

  for(i in 1:nrow(snps)){
    annots_i <- which((ann$Chrom == snps$CHROM[i]) &
           (ann$start < snps$POS[i]) &
            (ann$stop > snps$POS[i]) &
            ann$strand == snps$strand[i])
    if(length(annots_i) == 1){
      snps$SOFA[i] <- ann$annot[annots_i]
    } else if(length(annots_i) > 1){
      snps$SOFA[i] <- paste(sort(unique(ann[annots_i,]$annot)), collapse = ":")
    } else {
      snps$SOFA[i] <- "NC"
    }
  }

# create "neutral" vcf
```

```
col.vcf.nc <- col.vcf.good[which(snps$SOFA == "NC"), ]
col.vcf.cd <- col.vcf.good[which(snps$SOFA != "NC"), ]

write.vcf(col.vcf.nc, file = "~/Documents/Columbine/Data/GBS_Data/AQFO_snps_nc.vcf")
write.vcf(col.vcf.cd, file = "~/Documents/Columbine/Data/GBS_Data/AQFO_snps_cd.vcf")
```

References

Browning, Brian L., and Sharon R. Browning. 2016. "Genotype Imputation with Millions of Reference Samples." *The American Journal of Human Genetics* 98 (1): 116–26. https://doi.org/10.1016/j.ajhg.2015.11.020.

Maruki, Takahiro, and Michael Lynch. 2017. "Genotype Calling from Population-Genomic Sequencing Data." *G3: Genes, Genomes, Genetics* 7 (5): 1393–1404. https://doi.org/10.1534/g3.117.039008.