Project 3: Comparison of variant calling by `freebayes` and `mpileup` from high-throughput sequencing of Burbot (*Lota lota*)

Jason Moggridge 1159229

2021/04/19

Introduction

The analysis of genetic variants such as single-nucleotide polymorphisms (SNPs) and insertions/deletions (indels) by high-throughput sequencing is a central objective in studies of quantitative-trait loci, genome-wide associations, and population structure, among others. These polymorphic sites are also of practical interest in clinical genomics, *eg.* to discover disease-causing mutations (Cornish and Guda 2015), and in agriculture, where SNPs can be used as genetic markers to guide breeding of crops (Yao et al. 2020) and livestock (Ni et al. 2015). As the cost of sequencing has decreased and computational resources have improved, these techniques have become increasingly popular.

The identification of polymorphic sites from sequencing data, known as variant calling, is performed after alignment of reads to a reference genome. Because high-throughput sequencing and read-mapping are noisy, it is challenging but essential to differentiate true variants from artefacts, particularly in low-coverage data (Nielsen et al. 2011). Many variant calling tools (`bcftools`, `GATK`, `freebayes`, *etc.*) have been created for this classification task, though these differ in their error correction processes and statistical models for inferring genotypes. Further filtering of variants is generally applied after calling, and how this is performed depends on the nature of the study.

Both `bcftools mpileup/call` and `freebayes` apply Bayesian models to estimate genotype posterior probabilities, though they differ in how this is implemented. `bcftools` uses a two step process where `mpileup` first creates the 'pileup' of all reads at variant sites, and then `call` is used to infer genotypes at these (Li 2011; Danecek et al. 2021). This implementation has the advantage of being very fast. `freebayes` a uses different approach to inferring haplotypes in that it does not use the alignment but the full reads (Garrison and Marth 2012; "How to Freebayes - Erik Garrison's Blog" 2015).

I compared variant calling by two tools: `bcftools mpileup/call` and `freebayes`. The results were compared in terms of the number of SNPs, the overlap between sets, the depth at each position, and their minor allele frequencies. Previous comparisons of variant callers have often found low concordance between tools and issues with sensitivity (O'Rawe et al. 2013; Cornish and Guda 2015). However, others have reported high concordance and accuracy in their evaluation (Hwang et al. 2015); the best method is likely dependent on nature of the sequencing data (*eg.* whole-genome or exome, RAD-seq, RNA-seq, *etc.*), the sequencing depth, and the type of variants of interest (*eg.* SNPs or indels).

Methods

All files can be found on `cedar` *in* */scratch/jmoggrid/Project3/*.

Dataset

I used data from project #2 in this work, specifically the reads mapped to the burbot reference with the `bowtie2` aligner. First, I needed to convert my `.sam` files to `.bam` format and index them, as well as adding read groups using `picard` (necessary for `freebayes`):

```
#!/bin/bash
# to convert .sam to .bam, then sort & index
module load samtools
for file in bam/*.sam; do
  name=`echo $file | sed 's/\.sam//'`
  samtools view -b -S -o $name.bam $file
  samtools sort $name.bam -o $name.sorted.bam;
  samtools index $name.sorted.bam
done

# add readgroups and re-index; show output to verify
module load picard
group=1
for file in bam/*.sorted.bam
do
  label=`echo $file | sed 's/.sorted.bam//'`
  java -jar $EBROOTPICARD/picard.jar AddOrReplaceReadGroups I=$file \
    O=$label.rg.sorted.bam RGID=$group RGLB=lib1 RGPL=illumina RGPU=unit1 RGSM=$label
  samtools index $label.rg.sorted.bam
  echo "group number: $group; file: $file; read group:"
  samtools view -H $label.rg.sorted.bam | grep "@RG"
  group=$((group+1))
done
```

`bcftools mpileup` and `freebayes` variant calling

I used `bcftools mpileup/call` (referred to as `mpileup`) and `freebayes` to call variants from the ten burbot alignments. For `mpileup`, I added arguments to include annotations for read depth and allelic depth (-a DP,AD) and to specify Illumina as the sequencing platform (-P). For `call`, I specified the use of the newer multi-allelic caller (m), 'variants only' (v), to skip indels, and to provide the genotype quality scores (GQ). A summary of the raw SNP calls was obtained using `bcftools stats`.

```
module load nixpkgs/16.09  gcc/7.3.0 bcftools/1.9
# mpileup and variant calling
bcftools mpileup -Ou -a DP,AD -P ILLUMINA \
-f ./burbot_ref/GCA_900302385.1_ASM90030238v1_genomic.fna ./bam/*.rg.sorted.bam | \
bcftools call -mv --format-fields GQ --skip-variants indels -o  mpileup.call.vcf
bcftools stats mpileup.call.vcf > mpileup.call.stats
```

I called SNPs using `freebayes` with default settings except for the flag to provide annotations for genotype quality. Output was limited to SNPs with `vcffilter`.

```
module load freebayes/1.2.0
# create a list of 10 filenames to pass to as input to freebayes
> files.txt
for file in bam/*.rg.sorted.bam; do echo $file >> files.txt; done
```

```
# free-bayes calling for 10 files and restricting vcf output to SNPs w vcffilter
freebayes -f burbot_ref/GCA_900302385.1_ASM90030238v1_genomic.fna  \
  --genotype-qualities --bam-list files.txt | \
vcffilter -f "TYPE = snp" > freebayes.call.vcf
bcftools stats freebayes.call.vcf > freebayes.call.vcf
```

Filtering variant calls

I used the `vcftools` library to filter variant calls. Biallelic SNPs (`--min-alleles 2 --max-alleles 2 --remove-indels`) were retained if they had a quality score of at least 20 (`--minQ 20`). Genotypes were retained at these sites if they had genotyping quality score of at least 10 (`--minGQ 10`). I found the vast majority of variants to be shared among all 10 individuals (data not shown), so I applied filters for the maximum and minimum minor allele frequencies, such that these would not be either zero or one (`-maf 0.01 -max-maf 0.99`). Any calls that did not receive a 'PASS' flag were discarded (`--remove-filtered-all`).

```
module --force purge; module load StdEnv/2020 vcftools/0.1.16

# filter mpileup calls
vcftools --vcf mpileup.call.vcf --minQ 20 --minGQ 10 \
  --min-alleles 2 --max-alleles 2 --maf 0.01 --max-maf 0.99 \
  --remove-filtered-all --recode --recode-INFO-all \
  --out mpileup.filter

# same thing for freebayes
vcftools --vcf freebayes.call.vcf --minQ 20 --minGQ 10 \
  --min-alleles 2 --max-alleles 2 --maf 0.01 --max-maf 0.99 \
  --remove-filtered-all --recode --recode-INFO-all \
  --out freebayes.filter
```

There was no need to exclude any individuals from the analysis, as none had more that 22.5% missing data in the filtered `mpileup` calls, while the proportions of missing data in `freebayes` calls was roughly half as large (from `--missing-indv`).

```
# check individuals' missing data
vcftools --vcf mpileup.filter.recode.vcf --missing-indv \
  --out mpileup.filter
vcftools --vcf freebayes.filter.recode.vcf --missing-indv \
  --out freebayes.filter
```

Summarizing SNP calls

I counted the biallelic SNP calls from each caller with `bcftools stats`.

```
module load StdEnv/2020 bcftools/1.10.2
bcftools stats mpileup.filter.recode.vcf | grep "number of SNPs:" | cut -f4
bcftools stats freebayes.filter.recode.vcf | grep "number of SNPs:" | cut -f4
```

The overlap between sets of SNPs was computed with `bcftools isec`.

```
# prepare files for isec by zipping and re-indexing
bgzip mpileup.filter.recode.vcf
bcftools index mpileup.filter.recode.vcf.gz
bgzip freebayes.filter.recode.vcf
bcftools index freebayes.filter.recode.vcf.gz
# Compute intersection of 2 files with isec
bcftools isec -p isec mpileup.filter.recode.vcf.gz freebayes.filter.recode.vcf.gz
# Get overlap data from isec into tsv format
> Overlap.vals
for file in isec/*.vcf
do
  bcftools stats $file | grep "number of SNPs:" | cut -f4 >> Overlap.vals
done
echo $'unique to mpileup\nunique to freebayes\nshared\nshared' > Overlap.names
paste  Overlap.names Overlap.vals > Overlap.tsv
sed -i '1s/^/SNPs\tCount\n/' Overlap.tsv
```

Using `vcftools`, I computed the allele frequency at each loci across individuals, as well as the depth at each locus summed across individuals.

```
# allele frequency
vcftools --gzvcf mpileup.filter.recode.vcf.gz --freq  --out mpileup.filter
vcftools --gzvcf freebayes.filter.recode.vcf.gz --freq --out freebayes.filter
# site depth sum
vcftools --gzvcf mpileup.filter.recode.vcf.gz --site-depth --out mpileup
vcftools --gzvcf freebayes.filter.recode.vcf.gz --site-depth --out freebayes
```

Statistics

I created figures and summary tables from the data generated above using the `R` language. *This code is presented in the supplementary section at the end of the manuscript.*

Results

From the ten burbots in our sample, mpileup called a total of 132,046 SNPs, 173 of which were multiallelic, and with a transition:transversion ratio (ts:tv) of 0.81; freebayes called a total of 106,973 SNPs, with 1761 being multiallelic, and ts:tv of 0.44 (table 1). After applying filters for these were reduced to 11,115 SNPs with a 0.87 transition:transversion ratio for mpileup calls and 6,722 SNPs with 1.09 transition:transversion ratio for freebayes calls. The sets of biallelic SNPs identified by each caller were compared (table 2): mpileup called 65.35 % more biallelic SNPs than freebayes. Of these, a large proportion (6,208) were shared between both sets, representing 55.85 % of mpileup calls and 92.35 % of freebayes calls. Additionally, 4907 SNPs were unique to mpileup and 514 unique to freebayes.

Table 1: Raw and filtered SNP counts and transition/transversion ratios for calls by mpileup and freebayes.

Caller	raw SNPs	raw ts/tv	post-filter SNPs	post-filter ts/tv
mpileup	132,046	0.81	11,115	0.87
freebayes	106,973	0.44	6,722	1.09

The distribution minor allele frequencies of SNPs from mpileup shows a U-shape with peaks at 0.05 and 0.5, whereas those from freebayes peak at 0.05 and are otherwise evenly distributed (fig.1A). The range of possible values is constrained due to the small sample size (some values can only be obtained with a specific amount of missing data; fig. 1B). The mean depth per locus across individuals is mostly in the range of 8x-30x, with both callers showing peaks at 25x and similar distributions (fig. 2A). The mean depth (per site, per subject) for mpileup was 17x and for freebayes 16x reads. Both callers have some extreme outliers in terms of site depth: the maximum depth (summed across individuals) for freebayes was 16,162 reads, while for mpileup it was only 2522 reads. The minimum site depth (summed across individuals) was 5 and 3 for freebayes and mpileup respectively and mpileup calls a greater proportion of low-depth sites in general (fig. 2A). Freebayes also had a greater mean depth for each individual (fig. 2B). Both tools show similar distributions of quality scores, though mpileup calls a greater proportion of transversions at low-quality loci than freebayes (fig. 2C).

Table 2: Summary of total, shared, and unique SNPs called by mpileup and freebayes (after filtering).

SNPs	Count
mpileup	11,115
freebayes	6,722
shared	6,208
unique to mpileup	4,907
unique to freebayes	514

Discussion

In this work I compared the sets of biallelic SNPs called by two tools, bcftools mpileup/call and freebayes, from alignments of burbot reads from 10 individuals. The raw SNP calls were filtered similarly for both sets based on quality scores, genotyping confidence scores, and minor allele frequency; most of the data (>90%) was discarded during filtering (table 1), with roughly one-third of calls having allele frequency of 0 or 1 (*ie.* all individuals were similar but differed from the reference). Unexpectedly, I found that mpileup called many more SNPs that freebayes, and retained nearly twice as many after filtering. This is in contrast to findings from Cornish and Guda (2015) who found that freebayes called more SNPs in human exome data, though both tools had similar-sized sets after filtering. The discrepancy between these findings does not seem to be explained by the slight differences in filtering (they filtered for read depth; I filtered for genotype quality), but could be related to differences between datasets (Hwang et al. 2015).

Interestingly, the sets of SNPs called by mpileup and freebayes showed a high degree of overlap, with over 90% of freebayes calls being shared with the larger mpileup set (table 2). This result is somewhat in agreement with Hwang et al. (2015), who found ~90% concordance between (Samtools) mpileup, freebayes, and GATK on the gold-standard Genome in a Bottle dataset. My results suggest then, that either mpileup has low specificity (calls variants that don't exist), or that freebayes has low sensitivity (misses true variants). This does not seem to be the case generally though, as Cornish and Guda (2015) suggest the opposite and found that a variety of tools had similarly low sensitivity on the GiaB gold-standard data.

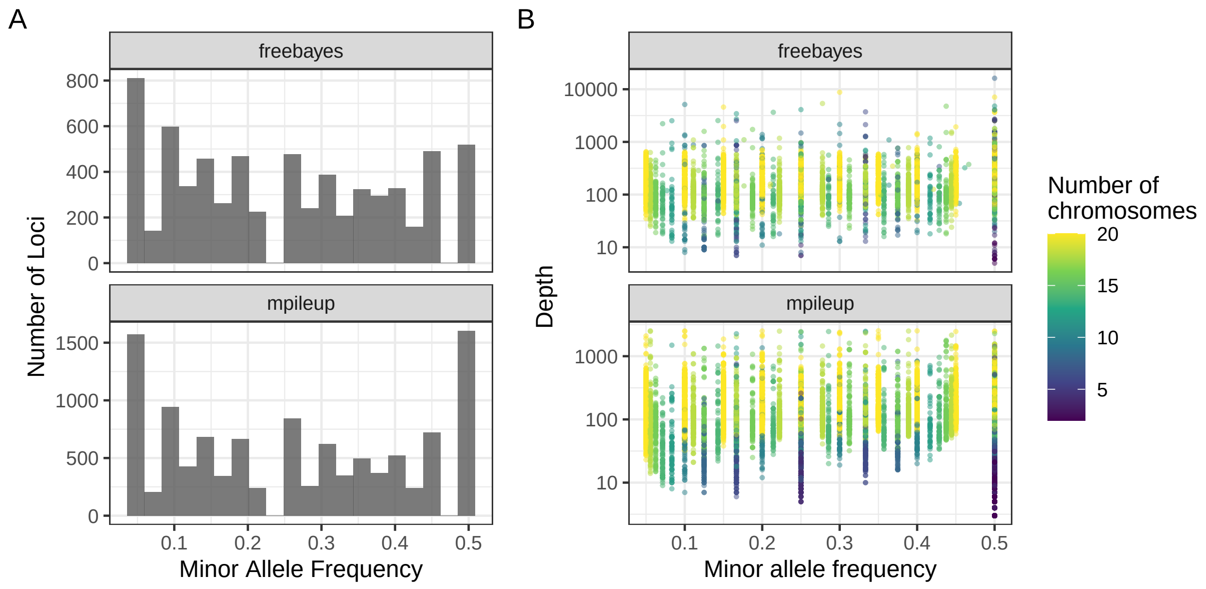

Figure 1: (A) Distribution of minor allele frequency of freebayes and mpileup SNP calls; SNPs with frequency = 0 or 1 were discarded in filtering (B) Scatterplot of loci (points) by allele depth and minor allele frequency, colored by the number of chromosomes counted. Due to the sample size (n=10), minor allele frequencies are constrained by the number of chromosomes at the SNP site; there appears to be no relationship between depth (ie. read coverage) and minor allele frequencies.

My analysis also revealed differences in the distribution of minor allele frequencies (MAF) and read depth at SNP loci (fig. 1). Specifically, mpileup shows a clear preference for SNPs with MAF of 0.5 and to a lesser degree 0.25 (relative to similar values). Both callers have a peak at low MAF (0.05) but otherwise mpileup has an even distribution from 0.10 to 0.5, when accounting for the constraint on these values by the small sample size (fig. 1B). The callers show differences in the read depth across loci (fig. 2A) and individuals (fig. 2B), as high-depth sites (>250x/subject) are hard-filtered by `mpileup`, and mpileup identifies many more low-depth sites (mean $<$ 7x/subject) that are ignored by freebayes. As such, the concordance between sets would likely increase if stricter filters were used to mitigate these biases.

Certain characteristics of the burbot data present challenges for variant calling. The highly-fragmented assembly means we are likely to miss many variants since they are not represented in the alignment. The relatively low coverage of the data means that a number of true variants are likely discarded because of large uncertainty in their classification. Further, burbot is not a model organism and does not have the same resources to develop useful prior probabilities for variant calling, *eg.* SNP databases for human variant calling (Nielsen et al. 2011), or benchmark datasets (such as GiaB) to validate pipelines (Zook et al. 2019).

In the present work, the small number of subjects means that the identification of uncommon variants is highly unlikely. If we interested in performing a population structure analysis of burbot, we should sequence more subjects and instead apply a filter for minor allele count instead of frequency to incorporate information from uncommon and rare variants (Linck and Battey 2019). Application of additional filters, *eg.* for strand-bias, site depth, and depth variation, would likely improve the accuracy of variant calling (O'Leary et al. 2018).

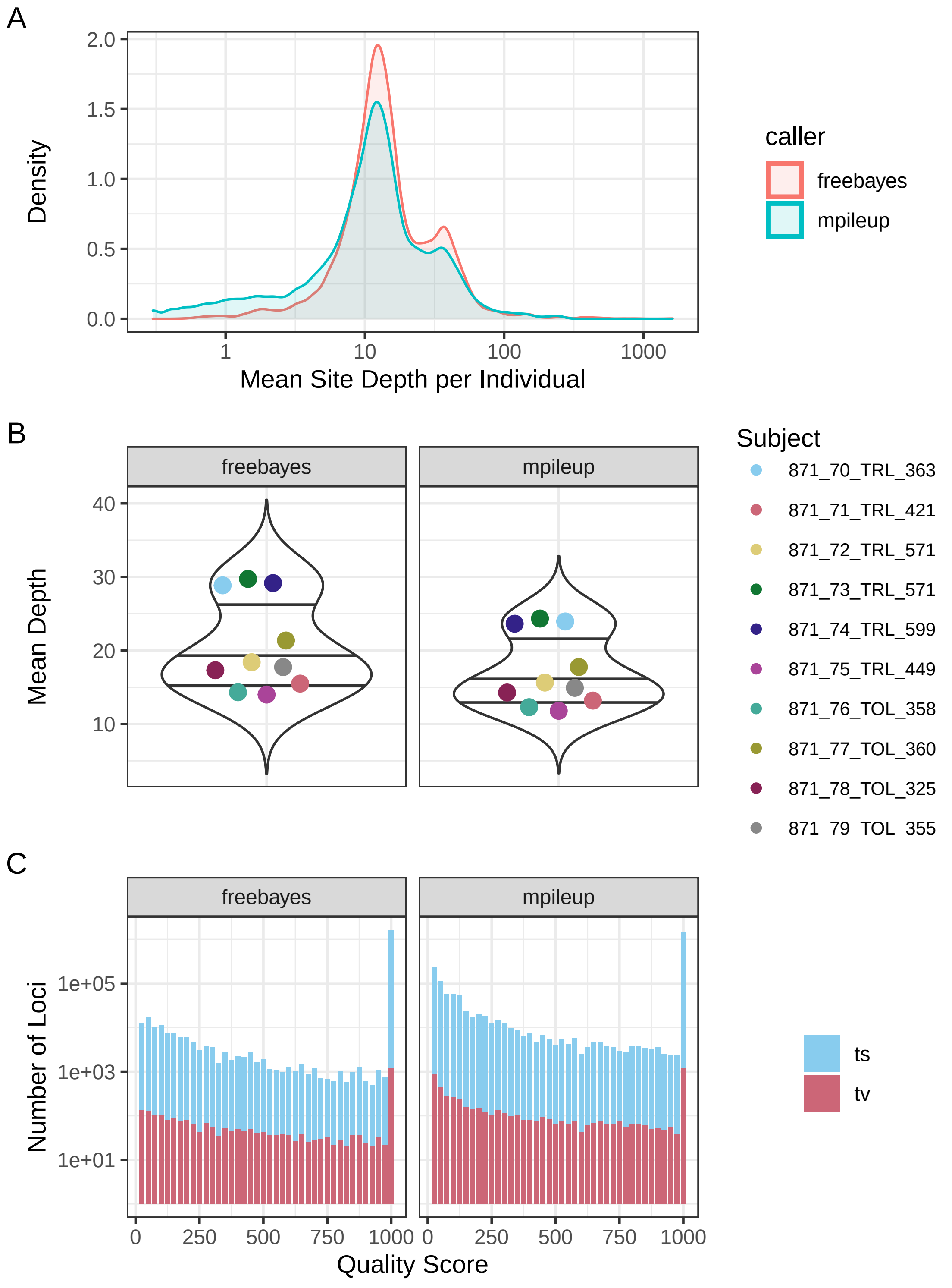

Figure 2: (A) Distribution of mean depth at SNP loci identified by freebayes and mpileup from 10 burbot subjects (nb: the x-axis is log-scaled).(B) Mean depth per locus for each individual. Violin plot lines indicate quartiles. (C) The quality score distributions for each set of SNP calls, separated into transitons (ts) and transversions (tv)

Supplemental

I also examined the quality score distribution over sites and the mean per site depth by individual. These were computed with `bcftools stats` and `vcftools` respectively, as follows (bash code):

```
## quality scores from bf
grep -E "Quality" mpileup.filter.recode.stats | cut -f1 --complement > mpileup.quality.stats
grep ^QUAL mpileup.filter.recode.stats | cut -f1 --complement >> mpileup.quality.stats
grep -E "Quality" freebayes.filter.recode.stats | cut -f1 --complement > freebayes.quality.stats
grep ^QUAL freebayes.filter.recode.stats | cut -f1 --complement >> freebayes.quality.stats
# Mean depth for each individual across all loci
vcftools --gzvcf mpileup.filter.recode.vcf.gz --depth --out mpileup
vcftools --gzvcf freebayes.filter.recode.vcf.gz --depth --out freebayes
```

R code for parsing results data

```
#! {R code}
library(tidyverse, quietly = T)
library(ggbeeswarm)
library(patchwork)
library(rcartocolor)
library(gridExtra)
library(grid)

# Overlap counts table; add totals for freebayes and mpileup
Overlap <- read_delim('./data/Overlap.tsv', '\t') %>%
  distinct() %>%
  bind_rows(
    tibble(SNPs = c('mpileup', 'freebayes'),
           Count = c(.[1,2][[1]] + .[3,2][[1]],
                     .[2,2][[1]] + .[3,2][[1]]))) %>%
  arrange(desc(Count))

# Allele frequencies
cnames <- c('chrom','pos', 'n_alleles','n_chr','a1','a2')
AlleleFreq <-
  bind_rows(
    read_tsv('./data/freebayes.filter.frq', col_names = cnames, skip = 1L) %>%
      mutate(caller = 'freebayes'),
    read_tsv('./data/mpileup.filter.frq', col_names = cnames, skip = 1L) %>%
      mutate(caller = 'mpileup')
  ) %>%
  mutate(allele = str_extract(a1, '[A-Z]+'),
         frequency = str_remove_all(a1, '[A-Z]+:'),
         allele2 = str_extract(a2, '[A-Z]+'),
         frequency2 = str_remove_all(a2, '[A-Z]+:')
  ) %>%
  mutate(across(contains('freq'), as.numeric)) %>%
  mutate(
    maf = ifelse(frequency > 0.5, 1 - frequency, frequency),
    minor_allele = ifelse(maf == frequency, allele, allele2)
  )
```

```
# Depth
cnames <- c('individual', 'n sites', 'mean_depth')
IndvDepth <- bind_rows(
    read_tsv('./data/mpileup.idepth', col_names = cnames, skip = 1L) %>%
      mutate(caller = 'mpileup'),
    read_tsv('./data/freebayes.idepth', col_names = cnames, skip = 1L) %>%
      mutate(caller = 'freebayes')
  ) %>%
  mutate(individual = str_remove_all(individual, 'bam.|.bowtie.burbot'),
         caller = as_factor(caller)) %>%
  arrange(individual)

# Site Depth
cnames <- c("chr", "pos", "allele_depth", "var_depth")
SiteDepth <- bind_rows(
  read_tsv('./data/mpileup.ldepth', col_names = cnames, skip = 1L) %>%
    mutate(caller = 'mpileup'),
  read_tsv('./data/freebayes.ldepth', col_names = cnames, skip = 1L) %>%
    mutate(caller = 'freebayes')
) %>% mutate(caller = as_factor(caller))

vcf_df <- SiteDepth %>%
  rename(chrom = chr) %>%
  left_join(AlleleFreq, by = c('chrom', 'pos', 'caller'))

cnames <- c('id', 'quality', 'n_snps', 'ts', 'tv', 'indels')
qual_df <-
  bind_rows(
    read_tsv("./data/mpileup.quality.stats", skip = 1L, col_names = cnames) %>%  mutate(caller = "mpileu
    read_tsv("./data/freebayes.quality.stats", skip = 1L, col_names = cnames) %>%  mutate(caller = "free
  ) %>%  mutate(caller = as_factor(caller))
```

R code for creating figures

```
## Figure 1L
# Allele frequency histogram (alleles stacked)
fig_allelefreq <-
  ggplot(AlleleFreq) +
  geom_histogram(aes(x = maf), bins = 20, alpha = 0.8) +
  facet_wrap(caller~., scales = 'free_y') +
  labs(x = 'minor allele frequency', y='loci') +
  theme_bw()

# The distribution of depth per-site summed across individuals
fig_sitedepth <-
  ggplot(SiteDepth, aes(x = allele_depth, fill = caller, color = caller)) +
  geom_density(alpha = 0.12) +
  geom_vline(aes(xintercept = mean(allele_depth), color = caller),
             lty = 2) +
  scale_x_log10() +
  labs(x = 'depth per locus', y = 'density') +
  guides(colour = guide_legend(override.aes = list(size=0.5, pch=15))) +
  theme_bw()

## Figure 2:

# Allele depth over maf, coloured by n chromosomes
fig_alleledepth_over_maf <-
  ggplot(vcf_df, aes(x=maf, y=allele_depth, color = n_chr)) +
  geom_point(size = 0.5, alpha = 0.7) +
  scale_y_log10() +
  facet_wrap(~caller) +
  scale_color_viridis_c() +
  labs(x = "Minor allele frequency",
       y = "Allele depth",
       color="Number of\nchromosomes") +
  theme_bw()

# The distribution of mean depth per site within individuals
set.seed(45)
fig_indvdepth <-
  ggplot(IndvDepth, aes(y = mean_depth, x = caller)) +
  geom_violin(alpha = 0.1, draw_quantiles = c(.25, .5, .75), trim = F) +
  geom_quasirandom(aes(color = individual), size = 3, width = 0.3) +
  scale_color_carto_d(palette = 2) +
  guides(colour = guide_legend(override.aes = list(size = 3, pch = 16))) +
  facet_wrap( ~ caller, scales = 'free_x') +
  labs(y = 'mean depth per locus', x = '', color = 'Individual') +
  theme_bw() +
  theme(axis.text.x = element_blank(),
        axis.ticks.x = element_blank(),
        legend.position = 'right')

# Quality score distribution; fill bars with type: ts or tv
fig_quality_dist <- qual_df %>%
  mutate(QUAL = round(quality/25)*25) %>%
  group_by(QUAL, caller) %>%
  summarise(n_snps = sum(n_snps),
            ts  = sum(ts),
            tv = sum(tv)) %>%
  pivot_longer(cols = c(ts,tv)) %>%
  ggplot(aes(x = QUAL, y = value, fill = name)) +
  geom_col(width = 19) +
  facet_wrap(~caller) +
  scale_fill_carto_d(palette = 1, direction = -1) +
  labs(x="quality score", y = "loci", fill = NULL) +
  scale_y_log10() +
  theme_bw()
```

R code to output tables and figures

```
## table 1
tribble(
  ~ Caller, ~ `raw SNPs`, ~ `raw ts/tv`, ~ `post-filter SNPs`, ~ `post-filter ts/tv`,
  "mpileup", 132046, 0.81, 11115, 0.87,
  "freebayes", 106973, 0.44, 6722, 1.09
  ) %>%
  pander::pander(caption = "Raw and filtered SNP counts and
                 transition/transversion ratios for calls by mpileup
                 and freebayes.")
### fig 1
(fig_allelefreq + fig_sitedepth) &
  plot_layout(guides = 'keep') &
  plot_annotation(tag_levels = 'A')
## table 2
Overlap %>%
  pander::pander(caption = "Summary of total, shared, and unique SNPs
                 called by mpileup and freebayes (after filtering).")

### fig 2
(fig_alleledepth_over_maf / fig_indvdepth / fig_quality_dist) &
  plot_annotation(tag_levels = 'A')
```

References

Cornish, Adam, and Chittibabu Guda. 2015. "A Comparison of Variant Calling Pipelines Using Genome in a Bottle as a Reference." https://www.hindawi.com/journals/bmri/2015/456479/.

Danecek, Petr, James K Bonfield, Jennifer Liddle, John Marshall, Valeriu Ohan, Martin O Pollard, Andrew Whitwham, et al. 2021. "Twelve Years of SAMtools and BCFtools." *GigaScience* 10 (2): giab008. https://doi.org/10.1093/gigascience/giab008.

Garrison, Erik, and Gabor Marth. 2012. "Haplotype-Based Variant Detection from Short-Read Sequencing." *arXiv:1207.3907 [q-Bio]*, July. http://arxiv.org/abs/1207.3907.

"How to Freebayes - Erik Garrison's Blog." 2015. https://ekg.github.io/2015/12/08/How-to-freebayes.

Hwang, Sohyun, Eiru Kim, Insuk Lee, and Edward M. Marcotte. 2015. "Systematic Comparison of Variant Calling Pipelines Using Gold Standard Personal Exome Variants." *Scientific Reports* 5 (1): 17875. https://doi.org/10.1038/srep17875.

Li, Heng. 2011. "A Statistical Framework for SNP Calling, Mutation Discovery, Association Mapping and Population Genetical Parameter Estimation from Sequencing Data." *Bioinformatics* 27 (21): 2987–93. https://doi.org/10.1093/bioinformatics/btr509.

Linck, Ethan, and C. J. Battey. 2019. "Minor Allele Frequency Thresholds Strongly Affect Population Structure Inference with Genomic Data Sets." *Molecular Ecology Resources* 19 (3): 639–47. https://doi.org/https://doi.org/10.1111/1755-0998.12995.

Ni, Guiyan, Tim M. Strom, Hubert Pausch, Christian Reimer, Rudolf Preisinger, Henner Simianer, and Malena Erbe. 2015. "Comparison Among Three Variant Callers and Assessment of the Accuracy of Imputation from SNP Array Data to Whole-Genome Sequence Level in Chicken." *BMC Genomics* 16 (1): 824. https://doi.org/10.1186/s12864-015-2059-2.

Nielsen, Rasmus, Joshua S. Paul, Anders Albrechtsen, and Yun S. Song. 2011. "Genotype and SNP Calling from Next-Generation Sequencing Data." *Nature Reviews. Genetics* 12 (6): 443–51. https://doi.org/10.1038/nrg2986.

O'Leary, Shannon J., Jonathan B. Puritz, Stuart C. Willis, Christopher M. Hollenbeck, and David S. Portnoy. 2018. "These Aren't the Loci You'e Looking for: Principles of Effective SNP Filtering for Molecular Ecologists." *Molecular Ecology* 27 (16): 3193–3206. https://doi.org/https://doi.org/10.1111/mec.14792.

O'Rawe, Jason, Tao Jiang, Guangqing Sun, Yiyang Wu, Wei Wang, Jingchu Hu, Paul Bodily, et al. 2013. "Low Concordance of Multiple Variant-Calling Pipelines: Practical Implications for Exome and Genome Sequencing." *Genome Medicine* 5 (3): 28. https://doi.org/10.1186/gm432.

Yao, Zhen, Frank M. You, Amidou N'Diaye, Ron E. Knox, Curt McCartney, Colin W. Hiebert, Curtis Pozniak, and Wayne Xu. 2020. "Evaluation of Variant Calling Tools for Large Plant Genome Re-Sequencing." *BMC Bioinformatics* 21 (1): 360. https://doi.org/10.1186/s12859-020-03704-1.

Zook, Justin M., Jennifer McDaniel, Nathan D. Olson, Justin Wagner, Hemang Parikh, Haynes Heaton, Sean A. Irvine, et al. 2019. "An Open Resource for Accurately Benchmarking Small Variant and Reference Calls." *Nature Biotechnology* 37 (5): 561–66. https://doi.org/10.1038/s41587-019-0074-6.