SNP Discovery from Single and Multiplex Genome Assemblies of Non-model Organisms

Morin, Phillip A; Foote, Andrew D; Hill, Christopher M; Simon-Bouhet, Benoit; Lang, Aimee R; Louis, Marie

Methods in Molecular Biology

DOI:
10.1007/978-1-4939-7514-3_9

Published: 01/01/2018

SNP discovery from single and multiplex genome assemblies of non-model organisms.

Phillip A. Morin[1], Andrew D. Foote[2], Christopher M. Hill[3], Benoit Simon-Bouhet[4], Aimee R. Lang[1], Marie Louis[5]

[1] Southwest Fisheries Science Center, National Marine Fisheries Service, National Oceanic and Atmospheric Administration, 8901 La Jolla Shores Dr., La Jolla, CA 92037

[2] Molecular Ecology and Fisheries Genetics Laboratory, School of Biological Sciences, Bangor University, Bangor, Gwynedd, LL57 2UW, UK

[3] Department of Computer Science, University of Maryland, College Park, Maryland, 20742 USA

[4] Centre d'Etudes Biologiques de Chizé, CNRS-UMR 7372, 79360 Villiers-en-Bois, France

[5] Scottish Oceans Institute, University of St Andrews, East Sands, St Andrews, Fife, KY16 8LB, UK

Correspondence: Phillip.Morin@noaa.gov

Running Head: SNP discovery from NGS data.

i. Abstract:

Population genetic studies of non-model organisms often rely on initial ascertainment of genetic markers from a single individual or a small pool of individuals. This initial screening has been a significant barrier to beginning population studies on non-model organisms *(1, 2)*. As genomic data become increasingly available for non-model species, SNP ascertainment from across the genome can be performed directly from published genome contigs and short-read archive data. Alternatively, low to medium genome coverage from shotgun NGS library sequencing of single or pooled samples, or from reduced-representation libraries *(*e.g., capture enrichment; *see* **Ref**. *3)* can produce sufficient new data for SNP discovery with limited investment. We describe protocols for assembly of short read data to reference or related species genome contig sequences, followed by SNP discovery and filtering to obtain an optimal set of SNPs for population genotyping using a variety of downstream high-throughput genotyping methods.

ii. Key Words

Bioinformatics, short read archive, reference-guided assembly, single nucleotide polymorphism, non-model organisms

1. Introduction

Despite the increasing ease, decreasing costs and wide variety of methods for sequencing complete genomes *(4)*, there is a continuing need for finding and genotyping specified polymorphic sites, for projects ranging from whole genome association studies *(5)* to population genetics *(e.g., 6 and references therein)* and phylogenetics *(7, 8)* to identification of functional variants *(9, 10)*. Once appropriate SNPs are identified, genotypes from hundreds to thousands of individuals can be obtained through cost-effective directed genotyping methods that can make use of small amounts and/or degraded DNA *(6, 11)*.

Prior to highly-parallel "next-generation" sequencing (NGS) methods, Sanger sequencing of individual loci was a limiting step, especially when genomic data were not available for PCR primer design *(1, 2)*. With NGS technologies, generation of targeted-locus *(3, 12, 13)* or whole-genome sequence data is now relatively routine, yielding large amounts of genomic data that can be screened for sites that are variable in a single genome *(14)* or among individuals *(3, 15, 16)*. A widely used method of both detecting novel SNPs and obtaining genotypes directly from reduced-representation genomic libraries is the RADseq method *(17)*. This method typically uses sample sets for detecting thousands to tens of thousands of genomic SNPs from larger numbers of individuals *(*where high-quality DNA is available from population samples, e.g., *see* **Ref**. *18)*. Software and methods for SNP detection and GBS have been previously described *(e.g., 19, 20)*, and so we will not discuss these methods further.

In this chapter, we will focus on the bioinformatics necessary to screen for SNPs from single-individual genome sequences representative of a species of interest, and from low-coverage genome sequences from one or more individuals and a reference genome of the same or a related species. There are several large consortium genome sequencing projects producing genomic data at an ever increasing rate *(e.g., 21, 22)*, providing raw data for SNP discovery. Even when a complete, high-coverage genome assembly for a species of interest is not available, generating low-coverage genome data and aligning to a related species can provide relatively quick and inexpensive access to thousands of SNPs in a target species.

The methods in this chapter assume availability of a Unix or Linux computing system with sufficient capacity (memory and storage) to handle Gigabyte data sets. The primary software packages are all freely available, though some additional commercial software is recommended (but not necessary) for some analyses.

2. Materials

The methods in this chapter are based on the versions of software listed in Table 1. There is no one public software package that does all of the steps described below, so we have used multiple freely-available programs for data processing. We also describe several scripts that have been written using different programs (e.g., Python, R, Perl), so the programs to run these scripts are necessary. Software can change, so it is important to use the specified versions, or test newer versions to make sure that the commands are performing the same functions and options remain as stated. The best way to do this (and familiarize yourself with the software) is to read the help documents or guidelines for each software package and the specified commands.

The current versions of the scripts described below are available through GitHub at https://github.com/PAMorin/SNPdiscovery. Unless otherwise stated, software is implemented in the Red Hat Enterprise 6 (64 bit) Linux Server operating system. Commands and software should function equally on Unix or Linux, but have not been tested on both.

3. Methods

This chapter assumes that short-read sequence data (typically 50-150bp, though this includes reads up to approximately 500bp are produced by some sequencing platforms) are available in one of the forms below, and that the publicly available data have been quality checked to remove adapter sequences and poor-quality reads. However, we outline adapter trimming steps for raw unprocessed sequence data in section 3.6 for shotgun sequence data (random sequence fragments from genomic DNA libraries) generated de novo for SNP discovery. For our purposes, we will consider two types of data: Publicly available genome assembly data that has already been assembled into a draft genome, and shotgun sequence data. For the latter, the assembled genome scaffold for a related species can be used to assist assembly, or in the absence of a reference genome, de novo assembly can be used to generate contigs to serve as the reference, though in most cases this will result in shorter contigs and potentially lower quality assemblies *(23)*.

All Linux command lines are shown in Courier font, and are prefaced by a comment line with "#" at the beginning to describe the function of the subsequent command. This is to facilitate copy/paste of the commands into a text document or directly into the Linux interface.

3.1 Publicly available genome assembly and short read archive (SRA) data.

The benefits of a reference genome assembly include longer contigs (or scaffolds), quality-checked assemblies, repeat masking, and in some cases, annotation and chromosome assignment. These can all facilitate high-quality SNP discovery and selection based on additional knowledge of genome position and association with known genes or chromosomes. The NCBI web site is a good place to search for reference genomes (http://www.ncbi.nlm.nih.gov/genome/browse/), but the organization can be confusing and there are important issues to be aware of. As an example, browsing for the genus "*Orcinus*" results in one record for the genome of *Orcinus orca*, the killer whale. This is a member of the family Delphinidae and could be used as the reference for other species in the family, and even in other families of cetaceans, though potentially at the loss of coverage and some bias in genome region coverage due to divergence. The Genome overview page for *Orcinus orca* (www.ncbi.nlm.nih.gov/genome/14034) summarizes the project information, publications, summaries and links to the mitochondrial and nuclear genome information (under "Replicon Region"), and Genome

Region, with links to the assembled scaffolds. We recommend starting under the Summary, and reviewing the BioProjects links to review the projects contributing to the killer whale genome. These may include multiple biosamples (different animals) and data types. Every project is different, so these need to be reviewed to determine which project(s) has the appropriate assembled contigs/scaffolds and SRA data for re-assembly and SNP discovery. For this example there are 2 BioProject ID's, and the project data for them are shown in **Fig. 1a, b**.

Both BioProjects are based on the same, single biosample, and link to the same assembly details, but one links to the SRA experiments (the sequence read data). If there are multiple biosamples, it is important to determine which SRA files are from each individual if the goal is to use either pooled sample data or just a single individual for SNP discovery.

On the BioProject description web page, the Assembly link leads to the project summary. All of the assembly files can be obtained via ftp through the NCBI site: ftp://ftp.ncbi.nlm.nih.gov/genomes/. After connecting as a guest, you can navigate to the species directory and view the README file that describes each of the directories and files within. For our purposes, the important files are in the directory CHR_Un, which contains the "Chromosome directories", or "concatenated sequence data for scaffolds that have been assembled from individual GenBank records", in a variety of formats. The files ending in ".fa.gz" are contigs in compressed FASTA format, and the ones ending in ".mfa.gz" are repeat-masked contigs in compressed FASTA format. These can be downloaded by a variety of methods. Probably the most useful in a Linux environment is to use "wget", for example:

```
# transfer reference genome file (compressed FASTA)
wget ftp://ftp.ncbi.nlm.nih.gov/genomes/Orcinus_orca/CHR_Un/
oor_ref_Oorc_1.1_chrUn.mfa.gz

# decompress the file.
gunzip oor_ref_Oorc_1.1_chrUn.mfa.gz
```

```
# Index your reference using the faidx function in SAMtools
# to allow efficient random access to the reference bases.
# This creates a file with the same name, but ending with “.fai”.
samtools faidx oor_ref_Oorc_1.1_chrUn.mfa

# Create index files needed for BWA read alignment to the reference.
# This creates 5 files with the same file name but different suffixes.
bwa index oor_ref_Oorc_1.1_chrUn.mfa
```

File transfer may take several hours, depending on the size of the file and speed of connection. It is usually not advised to map reads to a repeat masked genome, as reads neighboring the masked repeats may not map, thereby reducing coverage around the repeats. However, if you have a reference genome with reasonably high coverage, the loss will be minimal relative to the amount of high-coverage genome data, and using the repeat-masked reference sequence will reduce the need for filtering out SNPs in repeats later.

The next step is to identify and download the short-read archive data.
Clicking on the SRA link in the “Project Data” table on the BioProject page (e.g., “7” under “Number of Links” in the killer whale example) produced a list of the SRA experiments, each of which may have ≥1 sequence run files. You can look through these on the web site to identify the type of sequence (e.g., Illumina HiSeq 2000, paired-end) and the size of the individual files. Once you know which files you want to use, you can download individual files from the SRA portal http://sra.dnanexus.com. At the site, you can search for your species, select the appropriate study (if there is more than one), and click the “related runs” button to show all of the sequencing runs associated with that project. For this example, the project with accession SRP015826 has 7 experiments with 16 runs total, representing the 16 short-read archive files. The files can be selected by checking the boxes of those you want and using the “Download” button. This generates a web page with the download buttons for individual files, or you can select files and click the button for “download SRA URLs” to download a text file containing all of the URL

links to facilitate command-line transfer of the files using the “wget” command in a Linux environment:

Example SRA URLs file: download_sra_urls.txt (for the *Orcinus orca* project):

```
ftp://ftp-trace.ncbi.nlm.nih.gov/sra/sra-instant/reads/ByRun/sra/SRR/SRR574/SRR574967/SRR574967.sra
ftp://ftp-trace.ncbi.nlm.nih.gov/sra/sra-instant/reads/ByRun/sra/SRR/SRR574/SRR574968/SRR574968.sra
ftp://ftp-trace.ncbi.nlm.nih.gov/sra/sra-instant/reads/ByRun/sra/SRR/SRR574/SRR574969/SRR574969.sra
(etc.)
```

```
# Transfer each of the files using “wget” and the URLs in the
# download_sra_urls.txt document:
wget ftp://ftp-trace.ncbi.nlm.nih.gov/sra/sra-
instant/reads/ByRun/sra/SRR/SRR574/SRR574967/SRR574967.sra
```

These files are often very large, and the combined files can be several hundred GB, so the transfer may take a long time and require a lot of storage space on your system. It is important to determine how much storage space will be needed prior to starting to be sure you have sufficient storage capacity for the raw files and the assemblies. Although this is difficult to estimate and depends on whether or not you delete file that are no longer needed (e.g., delete sam files after converting to bam format), you should expect to use about 20 times as much disk space as the size of your decompressed short-read archive fastq file(s).

SRA files are compressed binary files and need to be expanded into FASTQ files prior to use in assembly. This requires the fastq-dump function from the SRAtoolkit software. Files size will expand substantially, typically by 4-5x the original SRA file size.

```
# Convert each SRA files into FASTQ files
fastq-dump SRR574967.sra --split-3
```

The “--split-3” option indicates that the file contains paired-end reads and will be split into two fastq files, one for read 1 and a second for read 2 (e.g., in this case, SRR574967_1.fastq, SRR574967_2.fastq). If the file only contains single-read data, only a single file will be generated, so it is recommended that this option is always used to ensure data are correctly parsed into files.

The FASTQ file may be too large to be processed when only one SRA file is available on the SRA portal for a high coverage genome. Fastq-splitter can be used to divide the large file in several smaller files that are easier to handle. After downloading the program, you need to make sure that the perl script is set as an executable.

```
# Command to check if the perl script is set to be executable
chmod +x fastq-splitter.pl
# Split the file in 6 parts for both the forward and reverse reads
./fastq-splitter.pl --n-parts 6 --check SRAfileF.fastq
./fastq-splitter.pl --n-parts 6 --check SRAfileR.fastq
# The path to the perl script (fastq-splitter.pl) should be given.
# --n-parts indicates into how many files the large
# SRA file should be divided (6 in this example).
# --check does some verification of FASTQ format correctness.
```

3.2 Creating a de novo set of reference contigs:

A draft genome sequence is not necessary to align sequence reads for SNP discovery, but some reference is necessary. In the absence of a reference genome, a set of reference contigs can be created by de novo assembly of short-read data. The details of de novo assembly are not the focus of this chapter, but there are commercially available programs (e.g., CLC Genomics Workbench (CLCbio), Geneious (BioMatters Limited)) and freely available programs such as SOAPdenovo2 *(24),* DISCOVAR de novo (Broad Institute), SGA *(25)*, some of which are optimized for different operating systems or types of data (e.g., see list at http://omictools.com/genome-assembly-category). The goal of using these packages is to generate high-quality contigs, then use them as the reference sequences to re-align reads for SNP discovery. Once a set of de novo assembled contigs has been produced (in FASTA format), they can be treated the same as a reference genome sequence for read alignment and SNP discovery *(see* **Ref**. *23).*

3.3 Alignment of reads to reference contigs:

Alignment of reads to reference contigs is fairly straightforward, but requires multiple steps that can be confusing. There are multiple alignment programs, but one of the most frequently used and easiest to use is the Burrows-Wheeler Aligner *(*BWA; *26)*.

When there are multiple smaller FASTQ files for a single individual, they can be concatenated into a single file for each read direction. This is a Linux function:

```
# Concatenate forward read files.
cat filename1_read1.fastq filename2_read1.fastq > forward_reads.fastq
```

However, when the files are large (e.g., >10GB), running separate alignments of each FASTQ file (or pair of files for paired-end data) will improve speed by allowing parallel processes run on separate cores (*see* **Note 1**). The resulting alignments can be combined after conversion to the smaller BAM files (see below).

Although there are many options that can be altered to vary alignment parameters, the updated aligner "BWA-MEM" algorithm *(27)* is implemented in BWA and has been optimized for aligning short reads to a large genome such as human without the need to specify alignment parameters, and performs as well as or better than many other aligners with short read (100bp) and longer read-length data *(27)*.

Mitochondrial DNA can make up a significant portion of sequence reads, so removal of reads that map to the mitogenome can reduce file sizes and speed up subsequent analyses. A reference mitogenome FASTA file can typically be downloaded from GenBank for the target species or a closely related species, or the mitogenome sequence can often be assembled de novo from the NGS reads *(e.g., 28)*. Map the reads to the mitogenome with BWA, convert the SAM file to a BAM file and sort with SAMtools.

```
# Align reads (reads.fq) to indexed reference mitogenome
# (ref_mtgenome.fasta)
bwa index ref_mtgenome.fasta

bwa mem ref_mtgenome.fasta reads.fq > mapped_to_mt.sam
```

```
# Note, if there are paired-end reads, the two
# respective files are listed, e.g., reads_1.fq reads_2.fq.


# Convert the SAM file to a sorted BAM file.
samtools view -b -h mapped_to_mt.sam | samtools sort - mapped_to_mt_sorted
# output = “mapped_to_mt_sorted.bam”
```

The BAM file will also retain the reads that did not map to the mitogenome reference, i.e. the reads from the nuclear genome that we are interested in. These unmapped reads can be extracted to a new BAM file, and then converted to a FASTQ file.

```
# Copy reads that did not map to the mitogenome to a new file
# and convert from BAM file to FASTQ format.
samtools view -f4 -h mapped_to_mt_sorted.bam > unmapped.bam

samtools bam2fq unmapped.bam > reads1.fq
# For paired-end reads:
cat reads.fq| grep '^@.*/1$' -A 3 --no-group-separator > reads1.fastq
cat reads.fq | grep '^@.*/2$' -A 3 --no-group-separator > reads2.fastq
```

Align the unmapped (non-mtDNA) reads to the reference genome using BWA. BWA requires an indexed FASTA reference file (see ‘bwa index’ above). The output file is in SAM format.

```
# Map reads (reads1.fq) to the reference FASTA file (ref.fa):
bwa mem –t2 ref.fa reads1.fq > aln_se1.sam
# -t option = number of threads to use (default = 1).
```

The output SAM files are typically very large, and can be reduced to the binary form, or BAM file format, for subsequent analyses. The SAM files can then be deleted, as they are no longer needed.

```
#Convert SAM file to BAM file and sort BAM file.
```

```
samtools view -@1 -T ref.fa -b aln_se1.sam | samtools -@1 sort - aln_se1_sorted
# -@ indicates the number of threads to use. Default = 1. This
#    option is only available in samtools for the ‘view’ and
#    ‘sort’ functions.
# -b Output in the BAM format
# -T FASTA format reference file.
```

PCR duplicates should be removed using the rmdup function of SAMtools. This is very important, as PCR amplification can bias the distribution of reads across loci and alleles.

```
# Collapse clonal reads
samtools rmdup -S aln-se1_sorted.bam aln-se1_sorted_unique.bam
# NOTE: There is a bug in Samtools 1.2 that causes this to not work. Both earlier and later (v1.3) versions
have been reported to work.
```

If there are several BAM files for an individual (e.g., aln-se1_sorted_unique.bam, aln-se2_sorted_unique.bam, etc.), they need to be merged into one file (individual1.bam).

```
# Merge the files
samtools merge -@2 individual1.bam aln_se1_sorted_unique.bam aln_se2_sorted_unique.bam
aln_se3_sorted_unique.bam aln_se4_sorted_unique.bam
# -@2 indicates that 2 threads are used.
```

At this stage, the sorted BAM file contains both mapped and unmapped sequencing reads. You may therefore want to remove the unmapped sequencing reads to save on disk storage space and improve the efficiency of your downstream analysis. But beware that your unmapped reads may contain useful data, such as reads associated with the Y-chromosome if you mapped your reads to a female reference, or microbiome data, etc.

```
# Keep reads that have mapped to the reference genome with a qscore ≥30.
samtools view -bh -F4 -q 30 individual1.bam > individual1_q30.bam
```

If the sequence reads have been aligned to a related species rather than reference contigs from the same species, a new target-species reference needs to be generated from the merged alignment file, and the alignment process is started over. This results in the reads being mapped to contigs of the same species so that inferred SNPs are due to intra-specific variation rather than differences between the reference and target species sequences.

```
# Extract the consensus sequences of the target-species contigs
# from the alignment.
samtools mpileup -uf ref.fa individual1.bam | bcftools call -c | vcfutils.pl vcf2fq > target_consensus.fq
# Convert the output fastq file to a fasta file using SEQTK
# to use as the new reference sequence, and index using
# ‘bwa index’ as above.
seqtk seq -a target_consensus.fq > target_consensus.fasta
```

3.4 Pipeline 1: SNP discovery from high-coverage genome assembly.

SNP discovery from alignment files (BAM) with medium to high average coverage (>10×) uses depth and quality criteria, and extracts SNPs with flanking sequence data for assay design (e.g. for AmpliSeq, GTseq, TaqMan, or capture-enrichment GBS). We describe a simple SNP discovery pipeline that uses the variant finder function of BCFtools *(29)* to scan an alignment (BAM file) for SNPs that meet specified criteria. The minimum and maximum coverage criteria are dependent on the average coverage of the genome alignment. When coverage is moderate (i.e., 10–30×), we set the minimum depth of coverage for a SNP at 10 reads, and a maximum depth of 100 in order to ignore potentially collapsed duplicated sequence in the assembly. However, when the average depth of an alignment is high (i.e., >30×), the criteria should be changed to reflect the range from 1/3 to twice the average coverage (see “Estimate coverage” in **Section 3.8**).

The BAM file to be used for SNP discovery needs to be indexed for SAMtools.

```
# Index BAM file for SAMtools.
samtools index individual1_q30.bam
```

The SNP discovery pipeline is managed through a shell script (*see* **Appendix 1**) that performs several functions. The shell script is launched by executing the script in a directory that also must contain the reference FASTA file (previously indexed with samtools faidx), the alignment BAM file (sorted) and the SNP discovery Python script “Script2_generate_genotype_blocks.py” (*see* **Appendix 2**). The shell script takes as input the alignment BAM file (sorted), reference FASTA file, and the total desired bases of the flanking sequence of the SNP (by default 300 bp). The shell script takes objects in order, assigning them to aliases that are used to fill in the appropriate filenames into a series of commands. Execution of the shell script first runs samtools mpileup to compile the read bases, quality and coverage information at each site in the alignment, then calls SNPs from the pileup using BCFtools, using filters to exclude loci with depth of coverage below and above specified levels (set within the shell script; *see* **Appendix 1**). The custom Python script “Script2_generate_genotype_blocks.py” further filters the SNPs based on user-defined quality cut-off and flanking sequence length parameters, and outputs two results files (*see* **Note 2**).

```
# Execute shell script1 for reference FASTA file and alignment
# BAM file.
./Script1_SNP_call_filter.sh individual1_q30.bam ref.fa 300 output
# ‘300’ indicates the sequence length, specifying 150bp on either
# side of the target SNP.
# ’output’ represents the file name that will be assigned to the
# output files.
```

The output files from this script are a FASTA file containing the designated target sequences surrounding each SNP, and a genotype VCF file. The VCF file format is described in detail elsewhere (https://github.com/samtools/hts-specs). This file contains information about each SNP, starting with the contig ID (“chrom”), the SNP position, the reference and alternate alleles, a quality score, and a list of additional “info” about the SNP, including the depth of coverage (DP), various quality measures, the count forward and reverse reads for each allele used in variant calling (DP4), and the consensus quality (FQ). (*see* **Fig. 2**). The total number of reads in DP4 may be lower than DP because low-quality bases are not counted. Given one sample in the alignment, the consensus quality (FQ) is typically positive for heterozygous loci, and negative for homozygous loci (*see* more details at http://samtools.sourceforge.net/mpileup.shtml). Although it seems counterintuitive that there would be homozygous SNPs, they may be frequent in some data sets, especially if the reference sequence is different from the reads mapped to it (different species, subspecies, or even just a different individual fixed for the alternate allele), and when reads are mapped incorrectly to the reference. A negative FQ value may also indicate that one of the alleles is rare.

The overall depth and the count of each base at the SNP site can be useful for further filtering of SNPs based on the number of nucleotides at the SNP position and their frequency. For example, if the SNPs are from a single individual, you would expect approximately even distribution of 2 alleles at heterozygous SNPs. A SNP with DP=20 and DP4 = 0,0,11,0 (No. of forward ref alleles, reverse ref alleles, forward non-ref alleles, reverse non-ref alleles) would indicate that even though there is a total depth of 20, only 11 reads met the quality threshold, and all of them were forward reads of the non-reference allele, so this is not likely to be a high-quality candidate SNP. A SNP with DP=20 and DP4=7,0,13,0 would have close to the expected 50% each of 2 alleles and all 20 reads passed the specified quality filters.

3.5 Pipeline 2: SNP discovery from low-coverage multiplex shotgun data

SNPs can be called from low-coverage shotgun sequencing data (<1×) from multiple samples using genotype likelihoods, therefore taking uncertainty in base-calling into account *(30)*. This section describes read sequence data processing, genotype likelihoods estimation and a filtering pipeline for SNP discovery using this approach *(following 31)*.

Data requirement: FASTQ formatted sequence read data from multiple samples. As an example, one lane of Illumina HiSeq or NextSeq sequencing of around 30 individuals (genome size of 2.5 Gb) would be suitable for genome-wide SNP discovery *(31)*. In the latter study, mean sequencing read depth per site considering all the samples together was approximately 5×, but the majority of the sites had sequencing reads from just a small proportion of the individuals sampled, each sequenced at <1× coverage. We describe data processing for de novo generated read data (i.e., generated by the investigator specifically for SNP discovery) and unmasked reference sequences, followed by SNP calling that can be applied to either the assembled de novo sequence data or SRA data assembled to a masked reference genome from GenBank.

3.6 Sequence data processing

This section assumes that single- or paired-end read data have been de-multiplexed to separate reads by the index sequences of each sample in the multiplex pool. SRA files downloaded from GenBank have presumably been processed to remove adapters and filtered for quality. For data generated specifically for SNP discovery, these steps need to be done prior to mapping reads to the reference genome. For each FASTQ file, ADAPTER-REMOVAL *(*or another program, e.g., Trimmomatic; *32)* can be used to remove adapter sequences from the sequencing reads and remove sequence reads that are ≤30 bp or another length threshold following trimming.

```
# Remove adapter sequences.
AdapterRemoval --file1 ind1.fastq --basename ind1trimmed --minlength 30 --trimns --trimqualities --
minquality --gzip
# ind1.fastq is the input file and ind1trimmed the output file.
#  --trimns: Remove stretches of Ns from the sequencing reads in
#    the 5' and 3' end.
#  --trimqualities: consecutive stretches of low quality bases
#    (the quality threshold is set by minquality) are removed from
#    the 5' and 3' end of the reads. All bases with minquality or
#    lower are trimmed. If ‘trimns’ is also indicated, stretches of
#    low-quality bases and/or Ns are trimmed.
#  --minquality: quality threshold for the trimming of low quality
#    bases. Default is 2.
#  --minlength 30: reads that are > 30 bp are kept.
#  --gzip: the output file is compressed. There will be 2 output
#    files: one with the discarded sequencing reads
#    (ind1trimmed.discarded.gz) and one with the retained reads
#    (ind1trimmed.truncated.gz).
# --qualitybase: specifies the platform-specific base quality
#    score encoding, with the assumed default of Phred+33 used
#    by Illumina. AdaptorRemoval can also handle Phred+64 and
#    'Solexa'encoded quality scores, but this input should
#    be specified using the --qualitybase command option.
```

Reads in FASTQ format must then be mapped to the reference genome of the studied species or a related species, then converted to a sorted BAM file following the same steps outlined above. The reference can be repeat hard-masked, or repeats can be masked in the BAM file following mapping.

To hard mask the reference FASTA file to which reads are being mapped to, we use RepeatMasker. Repeat libraries must also be accessed from the RepBase database which is available through the GIRI Web site (http://www. girinst.org/repbase/index.html). Note that this requires opening an account, however, there is currently no charge for this for academic non-profit users.

```
# Mask repeats
RepeatMasker genome.fa -nolow -norna -specie Cetartiodactyla
# genome.fa is the input reference fasta file.
# -nolow specifies that only interspersed repeats are masked
#  and STRs and low complexity regions are retained.
# -norna  prevents the default masking of small RNAs
# -specie specifies the taxanomic group the query sequence belongs
#  to, in our examples the Cetartiodactyla, and then checks the
#  RepBase file for known repetitive elements for this taxonomic
#  group.
```

The output files include a masked copy of the input file, which contains the query sequences, with identified repeats and low complexity sequences masked and replaced with ‘N’s. A map file is also generated, in which coordinates in terms of contig, start and end position are given, which can then be used as a .bed file (*see* **below**). The advantage of this is that reads can be mapped to the unmasked reference and then removed, which increases coverage in regions adjacent to repetitive elements.

```
# Run BEDtools to remove repeated regions.
intersectBed -abam data.bam -b RepeatMask.bed -v | samtools view -h - | samtools view -bSh - >
data_noRepeats.bam
# The BAM file from which the repeated regions should be removed
# is given with the option -abam and the file containing the
# repeated regions with –b.
```

Follow the steps described above to download a reference genome file in FASTA format, and index it for processing with SAMtools and BWA. Align and remove mitogenome reads, align remaining reads to the reference, convert the resulting SAM file to BAM format, collapse clonal reads, merge multiple files for single samples (if necessary), and remove poor-quality alignments (MAPQ<30) as previously described (*see* **Section 3.3** above).

A "bamlist" file of the path to the BAM files for each individual needs to be created. It is a text file (data.bamlist) with a line for the path to each individual BAM file. If they are in the same directory as the ongoing analysis, then just the file names are listed.

```
individual1_q30.bam
individual2_q30.bam
# etc.
# NOTE: if the files are not in the current directory, the path must
# be specified (e.g., /path/individual1_q30.bam, etc.). See
# documentation for ANGSD for more details.
```

3.7 SNP calling using genotype likelihoods for low-coverage alignments

SNP calling is performed using genotype likelihoods that take uncertainty into account and can be estimated in ANGSD *(33)* from multiple samples as recommended by Nielsen *et al*. *(30)*. Uncertainty in SNP discovery in low coverage data can be the result of sequencing, base-calling, mapping and alignment errors. Briefly, genotype likelihoods are calculated using probabilistic methods and take the quality scores of the sequencing reads into account *(30)*. A base is considered as a SNP if the minor allele frequency is significantly different from 0 as inferred from a likelihood ratio test *(34)*. The threshold is set with the '-SNP_pval' option and $P < 0.000001$ is considered as conservative. The genotype likelihood is calculated using the SAMtools method with the '-GL1' option *(29, 33)* to infer the major and minor alleles with –doMajorMinor1 and to estimate major and minor allele frequencies using the EM algorithm with –doMaf 2 *(34, 35)*(*see* **Note 3**).

```
# Infer SNPs from the alignments listed in "data.bamlist"
# using genotype likelihoods in ANGSD.
angsd -GL 1 -out genolike_data -nThreads 2 -doGlf 2 -doMajorMinor 1 -SNP_pval 1e-6 -doMaf 2 -minQ
30 -bam data.bamlist
# -out indicates the output file (followed by the filename)
# -bam indicates the input file (followed by the file name),
#  which is the data.bamlist file created previously.
# -doGlf 2 creates a genotype likelihood beagle file.
# -nThreads indicates the number of threads.
# minQ 30 to set the minimum base quality (minQ) to 30
# Optional settings: -minMapQ 30 to set the minimum mapping
# quality (minMapQ) to 30 (if not filtered before).
```

Running ANGSD produces a compressed .mafs output file: genolike_data.mafs.gz. The file must be decompressed prior to use in the filtering steps (below).

```
#decompress .mafs file
gunzip genolike_data.mafs.gz
```

3.8 SNP filtering

A filtering pipeline is then applied to further avoid bias linked to next-generation-sequencing and low coverage data. The different filters are:

- <u>Filter 1: discarding SNPs in regions of poor mapping quality (MAPQ<30)</u>.

- <u>Filters 2 & 3: removing SNPs in regions of excessive coverage and high numbers of individuals</u>: Filter 2 removes SNPs that are in regions with twice the mean coverage. As this is a rather arbitrary threshold, we advise plotting and exploring the distribution of

coverage at this stage. Illumina shotgun sequencing data typically results in a Poisson distribution of different coverage depths, and so selecting the point at which the upper bound of depth of coverage starts to deviate from this distribution can be a more formal approach to set a threshold for determining regions of excessive coverage.
Filter 3 removes SNPs that are found (i.e. for which there is coverage) in a high number of individuals. The cut-off is defined by the upper tail of the distribution of the plot of the number of SNPs against the number of individuals. SNPs in the upper tail of the distribution are removed. The high coverage of these SNPs may be because they are in unmasked repeated regions, in nuclear mitochondrial DNA (NUMTs), or other mapping artifacts (i.e. paralogous loci). This filter is only recommended with ultra-low coverage data.

- <u>Filter 4: discarding SNPs that have an estimated minor allele frequency (MAF) < 0.05</u> as it is the smallest MAF that we could theoretically have with a cut-off of 10 individuals (threshold defined earlier but it may change depending on the data analyzed). However, estimations of genotype likelihoods should reduce the error rate. Additionally, rare variants are important for several applications such as the inference of demographic history based on the Site Frequency Spectrum and the estimation of several population genetic parameters *(36, 37)*. Hence, depending on the downstream analyses it may be worth keeping those SNPs.

<u>- Filter 5: Select only SNPs with flanking sequence of a specified length</u> (e.g., 150 bp on either side of the target SNP) and with no N's in the flanking region, to allow design of primers and/or probes.

<u>- Filter 6: SNP validation</u>. (Optional) This step is not necessary but if available, the identified SNPs could be compared to already existing genomic resources such as published high coverage genomes to provide further confidence in the discovered SNPs, and if they are from different geographical areas, to identify a set of globally shared variants.

- Filter 7: (Optional) Remove SNPs with excess heterozygosity. This step can only be conducted meaningfully when there are sufficient samples to infer a significant deviation from the expectations of Hardy-Weinberg equilibrium (HWE), and is useful in identifying putative SNPs that are the result of gene duplications, gene families, and repetitive regions.

- Filter 8: (Optional) Compare SNP locus sequences to reference databases (e.g., GenBank) to filter out loci that might match to non-target species, duplicated loci, gene families, and repeat regions.

The commands to run these 8 filtering steps are detailed below.

Filter 1 & 2 (removing SNPs in regions of poor mapping quality (MAPQ<30) and SNPs that are in regions with twice the mean coverage): Commands will be the same for filter 1 (poor mapping quality) and 2 (excessive coverage), the two scripts are provided and the differences between the two filters are highlighted below. The mapping quality filter is only needed if the regions of high mapping quality (MAPQ>30) were not previously selected using samtools view (*see* **Section 3.3** and **Note 4**).

First, coverage is estimated in angsd using the doDepth function. This function estimates the distribution of the depth of coverage for each individual as well as across all individuals.

```
# Estimate coverage
angsd -bam data.bamlist -doDepth 1 -out data -doCounts 1 -nInd 30 -minMapQ 30 -minQ 30
# Reads with a mapping quality (minMapQ) above 30 and nucleotide
#  qscore (minQ) above 30 are kept.
# -nInd corresponds to the number of sequenced individuals.
# -doDepth 1 function in angsd also requires the –doCounts 1
#  option to estimate coverage.
# -out indicates the output file, followed by the output filename.
```

Three output files are produced, including data.depthSample and data.depthGlobal. In data.depthSample each line represents one sample (i.e., one individual) and column 1 is the number of sites with a depth of 0×, column 2 the number of sites with a depth of 1×, etc. In data.depthGlobal the number of sites for each depth category is given across all samples. The mean depth across all samples can be calculated using this file (*see* **Note 5**). It should be noted that the sequencing depth for all sites and not only SNPs is provided here.

```
# generate distributions of depth and quality scores:
angsd -b data.bamlist -doQsDist 1 -doCounts 1 -maxDepth 100 -doDepth 1 -out bam.qc
# data.bamlist is the same text file list of bamfiles as used above.
# -doQsDist 1 counts the number of bases for each quality score
# -maxDepth 100 sets the maximum depth at 100; sites covered at
#           depth >100 are combined, which may produce a peak at 100
#           for samples with higher coverage.
# -out indicates the output file, followed by the output file name.
```

This command produced 4 output files: bam.qc.arg, bam.qc.depthGlobal, bam.qc.depthSample and bam.qc.qs. These files can be plotted using the R script Script10_plotQC.R (*see* **Appendix 12**). The script creates a pdf file of the plotted output, and the file “bam.qc.info” with the q-score and global depth data.

As an example, we consider here that the mean coverage across all samples is 5× and that regions with coverage >10× are potential unmasked repeated regions or mapping artifacts. Regions of poor mapping quality (Q<30) and excessive coverage (>10×) are detected using the CALLABLELOCI tool in GATK. Running CALLABLELOCI involves some data formatting.

```
# Create a dictionary file on the initial reference using Picard tools.
java -jar path_to/picard.jar CreateSequenceDictionary R= ref.fa O= ref.dict
# The output file must have the same stem name as the reference
# file so that it can be used by GATK based on the reference file name.
```

If using low coverage data it may be best advised to merge individual BAM files into one consensus sequence to better identify putative repetitive regions or other mapping artifacts.

```
# merge individual bam files
samtools merge data.bam individual1_q30.bam individual2_q30.bam individual3_q30.bam #etc.

# The read groups are added to the header using Picard tools.
java -jar -Xmx10g path_to/picard.jar AddOrReplaceReadGroups INPUT= data.bam OUTPUT=
data_withheader.bam SORT_ORDER=coordinate RGID=sample RGLB=sample RGPL=illumina
RGSM=sample RGPU=name CREATE_INDEX=true
# -Xmx10g; defines the maximum memory size for Java, e.g. 10g = 10GB
# Sort_order: to order the output file (if not specified the order
#           in the output file is the same as in the input file)
# RGID: Read Group ID Default value: 1
# RGLB: Read group library required
# RGPL=illumina, can be changed for other NGS platforms
# RGSM: Read group sample name required
# RGPU: Read group platform unit (e.g. run barcode) required
# CREATE_INDEX: create a BAM index when writing a coordinate-sorted
#           BAM file
```

CallableLoci in GATK can then be run to identify the regions of poor mapping quality (MAPQ<30) and excessive coverage (>2× mean coverage).

```
# Run CallableLoci in GATK.
java -jar path_to/GenomeAnalysisTK.jar -T CallableLoci -R ref.fa -I data_withheader.bam  --maxDepth 10
-mmq 30 -summary data_callable.summary -o data_callable.bed
# -T = Name of the tool to run.
# -R = The reference genome against which the sequence data was
#   mapped. The GATK requires an index file (samtools faidx) and
#   a dictionary file accompanying the reference. They need to have
#   the same filename, ending in .dict for the dictionary file and
#   .fai for the indexed file.
# -I = Input file containing sequence data (BAM or CRAM).
# --maxDepth corresponds to twice the mean coverage (global). In this
#   example the mean coverage is 5×.
# -mmq: The minimum mapping quality (filter 1: regions with a
#   mapping quality lower than this threshold will be considered
#   of poor mapping quality).
```

The lines of the data_callable.bed file look like the following (a few example lines have been extracted from different parts of the file):

```
JH472447        0       21      NO_COVERAGE
JH472447        21      65      LOW_COVERAGE
JH472447        65      80      CALLABLE
JH472447        80      133     LOW_COVERAGE
JH472447        133     150     CALLABLE
JH472447        601657  601658  LOW_COVERAGE
JH472447        601658  601711  POOR_MAPPING_QUALITY
JH472447        601711  601712  CALLABLE
JH472447        601712  601719  LOW_COVERAGE
JH472461        671807  671914  CALLABLE
JH472461        671914  671947  NO_COVERAGE
JH472461        671947  671951  LOW_COVERAGE
JH472461        671951  671966  CALLABLE
JH472461        671966  671996  EXCESSIVE_COVERAGE
JH472461        671996  671997  CALLABLE
JH472461        671997  671999  LOW_COVERAGE
```

This file contains all sites. It indicates the contig, scaffold or chromosome depending on the available information, the start and end positions of the region and then the characteristic of the region (i.e., whether the region has no coverage, low coverage, poor mapping quality, excessive coverage or if the region is callable). For example on scaffold JH472447 bases from position 601658 to 601711 are in a region of poor mapping quality and on scaffold JH472461 bases from position 671966 to 671996 are in a region of excessive coverage.

Two similar R scripts are used to select independently the regions with poor mapping quality and excessive coverage.

```
# Source the script (Appendix 3) file using "Rscript" or
#  execute individual lines from the script in the R environment.
Rscript Script3a_Filter1_mapping_quality.R
# Script is written to use the GATK output file named
#   "data_callable.bed".
```

The “poor_mapping_quality.txt” output file looks like the following:

```
JH472447        6307    6341    POOR_MAPPING_QUALITY
JH472447        6437    6475    POOR_MAPPING_QUALITY
JH472447        516203  516207  POOR_MAPPING_QUALITY
JH472447        601658  601711  POOR_MAPPING_QUALITY
```

```
# Source the excessive coverage script (Appendix 4) file using
#  “Rscript” or execute individual lines from the script in
#   the R environment.
Rscript Script3b_Filter2_excessive_coverage.R
# Script is written to use the GATK output file named
#   “data_callable.bed”.
```

The “excessive_coverage.txt” output file look like the following:

```
JH472456        305636  305662  EXCESSIVE_COVERAGE
JH472461        129045  129049  EXCESSIVE_COVERAGE
JH472461        671966  671996  EXCESSIVE_COVERAGE
JH472465        237841  237878  EXCESSIVE_COVERAGE
```

Then, the SNPs that are in regions of poor mapping quality are removed from the genolike_data.mafs file (from the Genotype Calling section above) using a second R script (*see* **Script 4a below**) and the “poor_mapping_quality.txt” filter.

The genolike_data.mafs (from part 3.7) file looks like the following:

```
chromo  position  major minor unknownEM  pu-EM          nInd
JH472447        724             G       C        0.406400        2.160509e-07    5
JH472447        6226            C       A        0.378201        5.862932e-11    8
JH472447        599458  G       A       0.432863        1.945000e-12    4
JH472447        601678  A       T       0.301786        4.667339e-10    11
```

In the first column (i.e. “chromo”), the contig, scaffold or chromosome number is given (depending on the assembly level of the reference genome). The second column corresponds to the position on the scaffold, contig or chromosome. The major and minor alleles are given. “unknownEM” corresponds to the minor allele frequency. “pu-EM” is the p-value indicating that the minor allele frequency is significantly different from 0 (i.e. that the base is a SNP). “nInd” indicates the number of individuals for which there is coverage at each SNP.

The SNP at position 601678 on scaffold JH472447 in the .mafs file is in a region of poor mapping quality (JH472447, positions 601658-601711). The R script is used to filter this SNP out. All the other SNPs in these example lines are not in a region of poor mapping in the example lines of the “poor_mapping_quality.txt” filter given above and will be kept.

```
# Source the “remove_poor_mapping_quality” script (Appendix 5)
#   file using “Rscript” or execute individual lines from the
#   script in the R environment.
Rscript Script4a_Filter1_Remove_poor_mapping_quality.R
# This script requires the input files “genolike_data.mafs”
#   and “poor_mapping_quality.txt”.
# The output file is “SNPs_goodquality.txt”
```

A similar script is used the remove the SNPs in regions of excessive coverage. The SNPs in those regions are likely to be in a repeat region, NUMT or in a region with some other mapping artifact. The script is run on the output file of Filter 1 (SNPs_goodquality.txt) using the “excessive_coverage.txt” filter.

```
# Source the “remove_excessive_coverage” script (Appendix 6)
#   file using “Rscript” or execute individual lines from the
#   script in the R environment.
Rscript Script4b_Filter2_Remove_excessive_coverage.R
# This script requires the input files SNPs_goodquality.txt
#   and “excessive_coverage.txt”.
# The output file is “Good_coverage_SNPs.txt”
```

Filter 3: remove SNPs covered in an excessive number of individuals

SNPs that are covered in an excessive number of individuals are removed using an R script to further remove SNPs in potential repeat regions or mapping artifacts.
The number of individuals for which there is coverage for a given SNP is indicated in the last column of the .mafs file (*see* **above**). The cut-off is defined by the user after examining the upper tail of the distribution of the plot of the number of SNPs against the

number of individuals. SNPs in the upper tail of the distribution are discarded. Here as an example, a cut-off of 10 individuals was defined.

```
# Source the “excessive_individuals” script (Appendix 7)
#   file using “Rscript” or execute individual lines from the
#   script in the R environment.
Rscript Script5_Filter3_excessive_individuals.R
# Requires input file from Script4 “Good_coverage_SNPs.txt”
# The output file is “SNP_10ind.txt”
```

<u>Filter 4: remove SNPs with a MAF (Minor Allele Frequency) <0.05</u>

The R script to discard the SNPs that have a MAF less than 0.05 would be the same as for the number of individuals. The column that is used for the filtering is “unknownEM”.

```
# Source the “Remove_rare_SNPs” script (Appendix 8)
#   file using “Rscript” or execute individual lines from the
#   script in the R environment.
Rscript Script6_Filter4_Remove_rare_SNPs.R
# Requires input file from Script5 "SNP_10ind.txt"
# The output file is “SNPs_MAF_good.txt”
```

<u>Filter 5 (scripts 7 and 8): Select SNPs with specified length flanking sequences for primer/probe design.</u> This step only selects SNPs that have at least the specified length of sequence on either side of the SNP, without N’s.

```
# Scripts 7 and 8 (Appendix 9 and 10) must both be in the
#   target directory along with
# The output of Script6 (“SNPs_MAF_good.txt”) and the reference
#   FASTA file.
./Script7_SNP_call_filter_GATK.sh SNPs_MAF_good.txt ref.fa 300 output
# ‘300’ specifies the sequence length, specifying 150bp on either
#   side of the target SNP.
# Substitute the name of the reference sequence FASTA file for “ref.fa”
# Substitute a descriptive name for “output”. This will be the base
#   name for 2 output files, output.geno.fasta and output.geno.txt.
#   The latter file contains columns for SNP locus (“chromo” from GATK
#   output), position, major allele, minor allele, unknownEM (minor
#   allele frequency), pu.EM (probability of SNP), nInd (number of
#   individuals with data at SNP site).
```

Filter 6: optional validation step

This filtering step is optional in case one wants to compare the discovered SNPs with other datasets (for example a .mafs file obtained using a high coverage genome or coming from another geographical area). The SNPs that are shared between the datasets are identified.

```
# Source the “compare_SNP_datasets” script (Appendix 11)
#   file using “Rscript” or execute individual lines from the
#   script in the R environment.
Script9_Filter6_compare_SNP_datasets.R
# requires input files from filter Script8 (e.g., output.geno.txt)
#   and another .mafs file. These must be specified in the script
#   text for importing to “data1” and “data2”.
# The script assumes that the files do not have headers, and adds
#   them. If you compare to a .mafs file that already has a header
#   row, then the script must be altered so that the new headers
#   are not added.
```

Filter 7: optional check for excess heterozygosity

If SNP discovery has been conducted on a set of ≥10 samples (putatively from a single population), a further filter to exclude SNPs that exhibit excess heterozygosity (possibly indicating presence of duplicated genes or repeats), based on significant deviation from expectations of Hardy-Weinberg equilibrium (HWE) proportions and a negative F value. This should be done only if samples have sufficient coverage (typically ≥10×) to allow robust detection of heterozygote excess.

```
# Run genotype likelihood caller in ANGSD, using list of bam files in bamlist.txt.
angsd -GL 1 -out hwe_genolike_data -nThreads 5 -doGlf 2 -doMajorMinor 1 -SNP_pval 1e-6 -doMaf 2 -
HWE_pval <0.05 -bam bamlist.txt
```

```
# output columns: Chromo, Position, Major, Minor, hweFreq, Freq, F, LRT, p-value.
```

This process is the same as inferring SNP likelihoods in section 3.7 above (may take several days), except that the HWE filter is inserted to identify all SNP loci that deviate from HWE expectations at the given p-value. The output file.hwe.gz contains the loci that deviate significantly from HWE. The output can be sorted and all positive F values removed, then used to remove SNPs with significant negative F values (excess heterozygosity) from the final data set.

```
# Decompress the ANGSD output file
gunzip hwe_genolike_data.mafs.gz

# Remove "#" from the header of the SNP list from Filter 4 (above).
sed -i 's/#chromo/chromo/' SNPs_MAF_good.txt
# -i indicates to edit in place (same file)

# Remove excess heterozygosity SNPs from the good SNPs list.
# input files: SNPs_MAF_good.txt, SNPs_hweExcessHet.txt.
Rscript Script11_Filter7_Remove_HWEexcessHet.R
#output = good_hwe_SNPs.txt

# Re-add "#" to the header prior to running Filter 5 script to
# extract the SNP list and fasta file with flanking regions.
sed -i 's/chromo/#chromo/' good_hwe_SNPs.txt
```

Filter 8: optional filter based on BLAST comparison. The output of a BLAST search can identify sequences that match closely to non-target species (e.g., contaminants), repeat regions, mitochondrial or sex chromosomes and gene families. The output may be opened in a spreadsheet format and filtered or sorted based on the user’s search words.

```
#BLAST+ search of NCBI nucleotide (nt) database.
blastn -db nt -query SNPs_geno.fasta -out SNPs_geno_blast.out -max_target_seqs 1 -outfmt "6 qseqid
sseqid evalue length pident salltitles" -remote
```

3.9. Applications

This approach could be applied to any species for which a reference genome or a reference of a closely related species exists to map the reads. The main advantage of this method is that it results in the discovery of a large number of SNPs. As this approach does not allow simultaneous SNP discovery and genotyping, the identified SNPs could then be target-sequenced using custom-produced SNP arrays or target enrichment capture arrays for many applications in population genomics. These SNPs would be particularly useful for applications which need a large number of SNPs such as inferences of demographic history based on the site frequency spectrum *(38)* or inferences of selective sweeps *(39)*. Samples used for SNP discovery should ideally span the geographical range of populations that will be target-sequenced to avoid ascertainment bias *(2)*.

SNP discovery in the absence of population data can result in some false positives, where identified SNP loci are either the result of sequencing errors, or, more commonly, assembly errors that result in duplicated loci or repeat regions being combined into single assemblies. The effect of including false positives that are in fact monomorphic (false positives due to sequencing error) is relatively minor, resulting typically in a small portion of loci being uninformative. The effect of including false positives that fall in repeated regions can be much more significant, as highly repeated loci can heavily bias the read distribution in genotyping methods that rely on capture enrichment or amplification of groups of loci. **Fig. 3** shows the distribution of reads from a set of 384 SNP loci genotyped from multiplexed amplicons (lifetechnologies.com/ampliseqcustom) of 95 DNA samples from blue whales. The first set of 384 loci was obtained from a draft blue whale genome assembly that had not been masked for repeats. Seven of the SNPs represented 90% of the read coverage, resulting in too low coverage of the remaining loci for genotyping (*see* **Fig. 3a**). The SNP loci were screened for repeats using RepeatMasker as described above and replaced with SNP loci that did not show evidence of repeats. Genotyping of the second pool of 384 SNPs resulted in an almost 6x increase in average coverage of non-repeated loci and an even distribution of coverage across most loci (*see* **Fig. 3b**). We also recommend using BLAST to further screen for loci that either map to multiple loci or map to non-target species, e.g., bacterial or parasite DNA that could contaminate the sequence assembly.

4. Notes

1. If there are multiple sample files, running them sequentially through the same process can be facilitated by using shell scripts (e.g., Script1_SNP_call_filter.sh, Appendix 1). A shell script is simply a text document containing the commands that would be entered sequentially. For running the same commands on a set of sample files, the same command can be entered for each sample, with only the filenames (input and output) changed. The script is executed by typing './' before the script name in the directory where your script and input files reside. Permissions may need to be set appropriately to allow execution of shell scripts (see "chmod" in Linux commands, *see* **Table 2**).

2. Within the shell script text file, the parameters for minimum and maximum depth of coverage are set by the user, as determined for the average depth of coverage of the BAM alignment file. The script also calls the python script, which must be in the same directory, and the path to the python program may also have to be specified.

3. The filenames provided in these steps are used later in R-scripts, so changing the names will require changes in the R-scripts as well. This process is slow, and may take days to complete. It does not typically use >1 thread, though it may increase thread use as it adds sample files. A newly published wrapper for the program ANGSD *(40)* may make some of these steps easier or improve visualization, but have not yet been tested by us.

4. If the mapping quality filter is applied and results in an empty output file (i.e., SNPs with mapping quality <30 have all been previously filtered out), then the subsequent filter will produce an error. This can be circumvented by copying the first line of the "excessive_coverage.txt" file and pasting it into the empty "poor_mapping_quality.txt" file prior to running Script4a. This will simply remove the one of the excessive coverage SNPs and allow the script to proceed.

5. Calculating mean depth of coverage from the file data.depthGlobal: Open the file in Excel, insert the depth of coverage from 0 to 100 in row 1 (above the depth data). Use a function to calculate the mean depth: =SUMPRODUCT(B1:CW1*B2:CW2)/ SUM(B2:CW2). B1:CW1 contain the different depth of coverage from 0 to 100 and B2:CX2 the number of sites for each depth of coverage.

Acknowledgements

We are grateful to Lisa Komoroske for helpful comments on the manuscript. Blue whale DNA sequencing was generously provided by Tim Harkins and Clarence Lee, Life Technologies, Inc., and by Gerald Pao, Salk Institute for Biological Studies.

References

1. Aitken N, Smith S, Schwarz C, Morin PA (2004) Single nucleotide polymorphism (SNP) discovery in mammals: a targeted-gene approach *Mol Ecol* 13:1423-1431
2. Morin PA, Luikart G, Wayne RK, SNP Workshop Grp (2004) SNPs in ecology, evolution and conservation *Trends Ecol Evol* 19:208-216 doi:10.1016/j.tree.2004.01.009
3. Hancock-Hanser B, Frey A, Leslie M, Dutton PH, Archer EI, Morin PA (2013) Targeted multiplex next-generation sequencing: Advances in techniques of mitochondrial and nuclear DNA sequencing for population genomics *Mol Ecol Resour* 13:254-268 doi:10.1111/1755-0998.12059
4. Goodwin S, McPherson JD, McCombie WR (2016) Coming of age: ten years of next-generation sequencing technologies *Nature Rev Genet* 17:333-351 doi:10.1038/nrg.2016.49
5. Narum SR, Campbell NR, Meyer KA, Miller MR, Hardy RW (2013) Thermal adaptation and acclimation of ectotherms from differing aquatic climates *Mol Ecol* 22:3090-3097 doi:10.1111/mec.12240
6. Seeb JE, Carvalho G, Hauser L, Naish K, Roberts S, Seeb LW (2011) Single-nucleotide polymorphism (SNP) discovery and applications of SNP genotyping in nonmodel organisms *Mol Ecol Resour* 11 Suppl 1:1-8 doi:10.1111/j.1755-0998.2010.02979.x
7. Morin PA et al. (2015) Geographic and temporal dynamics of a global radiation and diversification in the killer whale *Mol Ecol* 24:3964-3979 doi:10.1111/mec.13284
8. Bryant D, Bouckaert R, Felsenstein J, Rosenberg NA, RoyChoudhury A (2012) Inferring species trees directly from biallelic genetic markers: bypassing gene trees in a full coalescent analysis *Mol Biol Evol* 29:1917-1932 doi:10.1093/molbev/mss086
9. Richards PM, Liu MM, Lowe N, Davey JW, Blaxter ML, Davison A (2013) RAD-Seq derived markers flank the shell colour and banding loci of the Cepaea nemoralis supergene *Mol Ecol* 22:3077-3089 doi:10.1111/mec.12262
10. Takahashi T, Sota T, Hori M (2013) Genetic basis of male colour dimorphism in a Lake Tanganyika cichlid fish *Mol Ecol* 22:3049-3060 doi:10.1111/mec.12120
11. Campbell NR, Harmon SA, Narum SR (2015) Genotyping-in-Thousands by sequencing (GT-seq): A cost effective SNP genotyping method based on custom amplicon sequencing *Mol Ecol Resour* 15:855-867 doi:10.1111/1755-0998.12357
12. Faircloth BC, McCormack JE, Crawford NG, Harvey MG, Brumfield RT, Glenn TC (2012) Ultraconserved elements anchor thousands of genetic markers spanning multiple evolutionary timescales *Systematic biology* 61:717-726 doi:10.1093/sysbio/sys004
13. Lemmon AR, Emme SA, Lemmon EM (2012) Anchored hybrid enrichment for massively high-throughput phylogenomics *Systematic biology* 61:727-744 doi:10.1093/sysbio/sys049

14. Eck SH, Benet-Pages A, Flisikowski K, Meitinger T, Fries R, Strom TM (2009) Whole genome sequencing of a single Bos taurus animal for single nucleotide polymorphism discovery *Genome Biol* 10:R82 doi:10.1186/gb-2009-10-8-r82
15. Pavy N, Gagnon F, Deschenes A, Boyle B, Beaulieu J, Bousquet J (2016) Development of highly reliable in silico SNP resource and genotyping assay from exome capture and sequencing: an example from black spruce (Picea mariana) *Mol Ecol Resour* 16:588-598 doi:10.1111/1755-0998.12468
16. Aslam ML et al. (2012) Whole genome SNP discovery and analysis of genetic diversity in Turkey (Meleagris gallopavo) *BMC Genomics* 13:391 doi:10.1186/1471-2164-13-391
17. Baird NA et al. (2008) Rapid SNP discovery and genetic mapping using sequenced RAD markers *PLoS One* 3:e3376 doi:10.1371/journal.pone.0003376
18. Foote AD, Morin PA (2016) Genome-wide SNP data suggests complex ancestry of sympatric North Pacific killer whale ecotypes *Heredity* doi:10.1038/hdy.2016.54.
19. Narum SR, Buerkle CA, Davey JW, Miller MR, Hohenlohe PA (2013) Genotyping-by-sequencing in ecological and conservation genomics *Mol Ecol* 22:2841-2847 doi:10.1111/mec.12350
20. Andrews KR, Good JM, Miller MR, Luikart G, Hohenlohe PA (2016) Harnessing the power of RADseq for ecological and evolutionary genomics *Nat Rev Genet* 17:81-92 doi:10.1038/nrg.2015.28
21. Koepfli KP, Paten B, Genome KCoS, O'Brien SJ (2015) The Genome 10K Project: a way forward *Annu Rev Anim Biosci* 3:57-111 doi:10.1146/annurev-animal-090414-014900
22. i5K Consortium (2013) The i5K Initiative: advancing arthropod genomics for knowledge, human health, agriculture, and the environment *J Hered* 104:595-600 doi:10.1093/jhered/est050
23. Card DC et al. (2014) Two low coverage bird genomes and a comparison of reference-guided versus de novo genome assemblies *PLoS One* 9:e106649 doi:10.1371/journal.pone.0106649
24. Luo R et al. (2012) SOAPdenovo2: an empirically improved memory-efficient short-read de novo assembler *Gigascience* 1:18 doi:10.1186/2047-217X-1-18
25. Simpson JT, Durbin R (2012) Efficient de novo assembly of large genomes using compressed data structures *Genome Res* 22:549-556 doi:10.1101/gr.126953.111
26. Li H, Durbin R (2009) Fast and accurate short read alignment with Burrows-Wheeler transform *Bioinformatics* 25:1754-1760 doi:10.1093/bioinformatics/btp324
27. Li H (2013) Aligning sequence reads, clone sequences and assembly contigs with BWA-MEM. arXiv:13033997v1 [q-bioGN]
28. Lounsberry ZT, Brown SK, Collins PW, Henry RW, Newsome SD, Sacks BN (2015) Next-generation sequencing workflow for assembly of nonmodel mitogenomes exemplified with North Pacific albatrosses (Phoebastria spp.) *Mol Ecol Resour* 15:893-902 doi:10.1111/1755-0998.12365

29. Li H (2011) A statistical framework for SNP calling, mutation discovery, association mapping and population genetical parameter estimation from sequencing data *Bioinformatics* 27:2987-2993 doi:10.1093/bioinformatics/btr509
30. Nielsen R, Paul JS, Albrechtsen A, Song YS (2011) Genotype and SNP calling from next-generation sequencing data *Nature Rev Genet* 12:443-451 doi:10.1038/nrg2986
31. Louis M et al. (*in prep*) High density, genome-wide SNP discovery in Northeast Atlantic bottlenose dolphins using ultra low coverage shotgun sequencing data
32. Bolger AM, Lohse M, Usadel B (2014) Trimmomatic: a flexible trimmer for Illumina sequence data *Bioinformatics* 30:2114-2120 doi:10.1093/bioinformatics/btu170
33. Korneliussen TS, Albrechtsen A, Nielsen R (2014) ANGSD: Analysis of Next Generation Sequencing Data *BMC Bioinformatics* 15:356 doi:10.1186/s12859-014-0356-4
34. Kim SY et al. (2011) Estimation of allele frequency and association mapping using next-generation sequencing data *BMC Bioinformatics* 12:231 doi:10.1186/1471-2105-12-231
35. Skotte L, Korneliussen TS, Albrechtsen A (2012) Association testing for Next-Generation Sequencing data using score statistics *Genetic Epidemiology* 36:430-437 doi:10.1002/gepi.21636
36. Nielsen R (2004) Population genetic analysis of ascertained SNP data *Human genomics* 1:218-224
37. Clark AG, Hubisz MJ, Bustamante CD, Williamson SH, Nielsen R (2005) Ascertainment bias in studies of human genome-wide polymorphism *Genome Research* 15:1496-1502 doi:10.1101/gr.4107905
38. Excoffier L, Dupanloup I, Huerta-Sanchez E, Sousa VC, Foll M (2013) Robust demographic inference from genomic and SNP data *Plos Genetics* 9:e1003905 doi:10.1371/journal.pgen.1003905
39. Chen H, Patterson N, Reich D (2010) Population differentiation as a test for selective sweeps *Genome Research* 20:393-402 doi:10.1101/gr.100545.109
40. Durvasula A, Hoffman PJ, Kent TV, Liu C, Kono TJ, Morrell PL, Ross-Ibarra J (2016) ANGSD-wrapper: utilities for analyzing next generation sequencing data *Mol Ecol Resour* doi:10.1111/1755-0998.12578
41. Schubert M, Lindgreen S, Orlando L (2016) AdapterRemoval v2: rapid adapter trimming, identification, and read merging *BMC Res Notes* 9:88 doi:10.1186/s13104-016-1900-2
42. DePristo MA et al. (2011) A framework for variation discovery and genotyping using next-generation DNA sequencing data *Nat Genet* 43:491-498 doi:10.1038/ng.806
43. McKenna A et al. (2010) The Genome Analysis Toolkit: A MapReduce framework for analyzing next-generation DNA sequencing data *Genome Research* 20:1297-1303 doi:10.1101/gr.107524.110
44. R Core Team (2015) R: A language and environment for statistical computing. R Foundation for Statistical Computing, Vienna, Austria

45. Li H et al. (2009) The Sequence Alignment/Map format and SAMtools *Bioinformatics* 25:2078-2079 doi:10.1093/bioinformatics/btp352

Tables:

Table 1: Required and optional software. The version numbers were tested. Later versions should be compatible, but sometimes changes are made that may not be compatible with the scripts and commands as we have presented them.

Software	Version	Reference or URL
AdapterRemoval	2.0	*(41)*
ANGSD	0.911	*(33)*
BCFtools	0.1.19	*(29)*
BEDtools	2.25.0	https://github.com/arq5x/bedtools2/releases
BWA	0.7.5a	*(26)*
BLAST+ (optional)	2.2.28+	https://blast.ncbi.nlm.nih.gov/Blast.cgi?CMD=Web&PAGE_TYPE=BlastDocs&DOC_TYPE=Download
Fastq-splitter (optional)	0.1.2	http://kirill-kryukov.com/study/tools/fastq-splitter/ (Perl script)
GATK	2.5-2	*(42, 43)*
JAVA	1.8.0	https://Java.com
Picard Tools	1.92	http://broadinstitute.github.io/picard/
PYTHON	2.6.6	https://www.Python.org
R	3.2.5	*(44)*
RepeatMasker (optional)	4.0.6	http://www.repeatmasker.org/RMDownload.html
SAMtools	1.2	*(45)*
SEQTK	1.1-r92	https://github.com/lh3/seqtk
SRAtoolkit	2.5.7	http://www.ncbi.nlm.nih.gov/Traces/sra/sra.cgi?view=software/

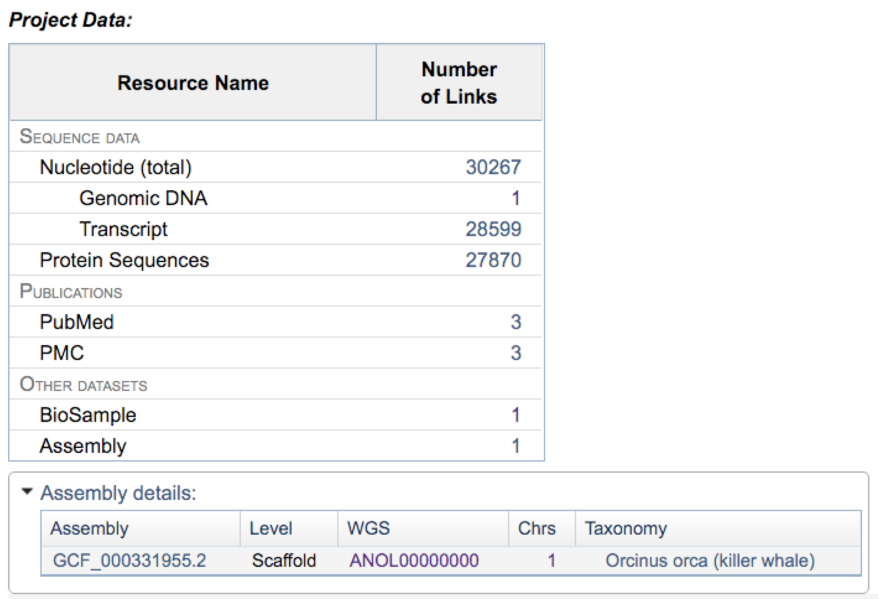
Table 2: Useful Linux commands:

wget	

nohup (when you can/can’t use this & (what it does and when you can/can’t use it))	
top	Typing “top” shows the current processes on the system, including how the percent CPU (i.e., 100% = 1 CPU) and percent of the total memory being used by each process. Exit with the keyboard combination “control” and “c”
df –h	shows current system status, including size, amount used, amount available, % used.
control-c	For a process that is active (i.e., not run in background mode), it can be cancelled with the keyboard combination “control” and “c”
kill	When a process is running in the background, it can be “killed” by using the “top” command to find the process ID (PID), then using typing “kill” and the PID.
Rscript	Run an R script (follow by specifying the path/name of the script).
Chmod 755	Change permissions on a file to allow everyone to read and execute the file, and the file owner is allowed to write to the file. Follow with file name.
grep	Useful for extracting text. For subsetting a fasta file from a list of loci: `grep -Fwf SNP_list.txt -A1 file.fasta | grep -v '^--$' >subset_file.fasta` # file.fasta contains a list of sequence name texts that will be found and selected from the fasta file, along with the following DNA sequence line for each locus.

Figures

a)



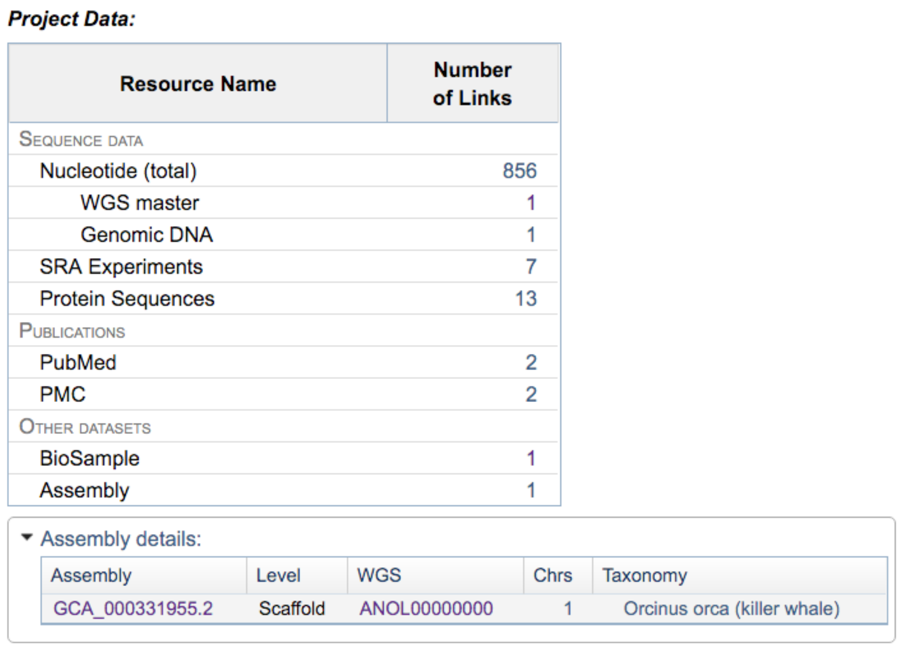
Project Data:

Resource Name	Number of Links
SEQUENCE DATA	
Nucleotide (total)	30267
Genomic DNA	1
Transcript	28599
Protein Sequences	27870
PUBLICATIONS	
PubMed	3
PMC	3
OTHER DATASETS	
BioSample	1
Assembly	1

Assembly details:

Assembly	Level	WGS	Chrs	Taxonomy
GCF_000331955.2	Scaffold	ANOL00000000	1	Orcinus orca (killer whale)

b)



Project Data:

Resource Name	Number of Links
SEQUENCE DATA	
Nucleotide (total)	856
WGS master	1
Genomic DNA	1
SRA Experiments	7
Protein Sequences	13
PUBLICATIONS	
PubMed	2
PMC	2
OTHER DATASETS	
BioSample	1
Assembly	1

Assembly details:

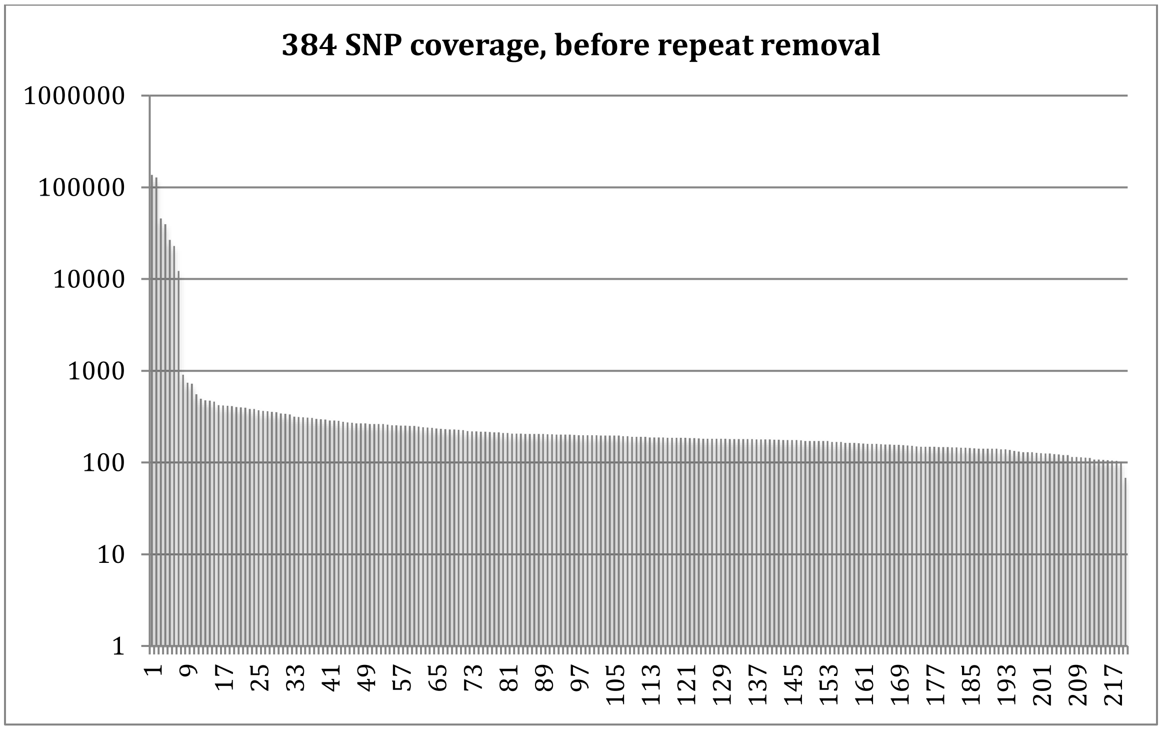
Assembly	Level	WGS	Chrs	Taxonomy
GCA_000331955.2	Scaffold	ANOL00000000	1	Orcinus orca (killer whale)

Fig. 1. Screen captures from GenBank genome projects web page for the killer whale genome (*Orcinus orca;* http://www.ncbi.nlm.nih.gov/genome/14034). Project data table for 2 BioProjects associated with this genome, a) PRJNA189949 and b) PRJNA167475

#CHROM	POS	ID	REF	ALT	QUAL	FILTER	INFO (DP=depth,	FORMAT
z0000161_contig_43910	270	.	C	T	50.0072	.	DP=10;VDB=0.748679;SGB=-0.511536;RPB=0.809011;MQB=0.924584;MQSB=0.974597;BQB=0.924584;MQ0F=0;AF1=0.5;AC1=1;DP4=3,3,2,1;MQ=60;FQ=53.0153;PV4=1,0.249156,1,1	GT:PL
z0000161_contig_43917	371	.	G	A	56.0072	.	DP=10;VDB=0.803107;SGB=-0.556411;RPB=0.89338;MQB=0.974597;MQSB=0.924584;BQB=0.730948;MQ0F=0;AF1=0.5;AC1=1;DP4=2,3,1,3;MQ=60;FQ=59.0154;PV4=1,0.224349,1,1	GT:PL
z0000161_contig_43964	1694	.	A	G	104.008	.	DP=10;VDB=0.381141;SGB=-0.616816;RPB=0.783809;MQB=1.00775;MQSB=1.00775;BQB=0.895781;MQ0F=0;AF1=0.5;AC1=1;DP4=2,2,2,4;MQ=60;FQ=79.0088;PV4=1,0.278763,1,0.237204	GT:PL
z0000161_contig_44252	358	.	C	T	87.0076	.	DP=10;VDB=0.277213;SGB=-0.590765;RPB=0.487298;MQB=0.974597;MQSB=0.924584;BQB=0.406082;MQ0F=0;AF1=0.5;AC1=1;DP4=2,2,4,1;MQ=60;FQ=89.9475;PV4=0.52381,0.0880959,1,0.0674986	GT:PL
z0000161_contig_44421	1951	.	T	C	77.0075	.	DP=10;VDB=0.66971;SGB=-0.556411;RPB=1.00775;MQB=1.00775;MQSB=0.916482;BQB=0.783809;MQ0F=0;AF1=0.5;AC1=1;DP4=4,2,3,1;MQ=60;FQ=80.0156;PV4=1,0.304798,1,1	GT:PL
z0000161_contig_44939	362	.	G	A	108.008	.	DP=14;VDB=0.651328;SGB=-0.616816;RPB=0.991701;MQB=1;MQSB=1;BQB=0.991701;MQ0F=0;AF1=0.5;AC1=1;DP4=6,2,5,1;MQ=60;FQ=111.016;PV4=1,0.470145,1,1	GT:PL

Fig. 2. Example of Variant Call Format (VCF) file format. Column headings are CHROM (contig ID), POS (SNP position), ID (sample IDs for some data types), REF (reference allele), ALT (alternate allele), QUAL (quality score), FILTER (filters that have been applied to the data), INFO (combined information types), FORMAT (genotype format codes), and additional genotype or sample file information. Additional details can be found at https://samtools.github.io/hts-specs/VCFv4.1.pdf.

a)


384 SNP coverage, before repeat removal
1000000
100000
10000
1000
100
10
1
1
9
17
25
33
41
49
57
65
73
81
89
97
105
113
121
129
137
145
153
161
169
177
185
193
201
209
217

b)

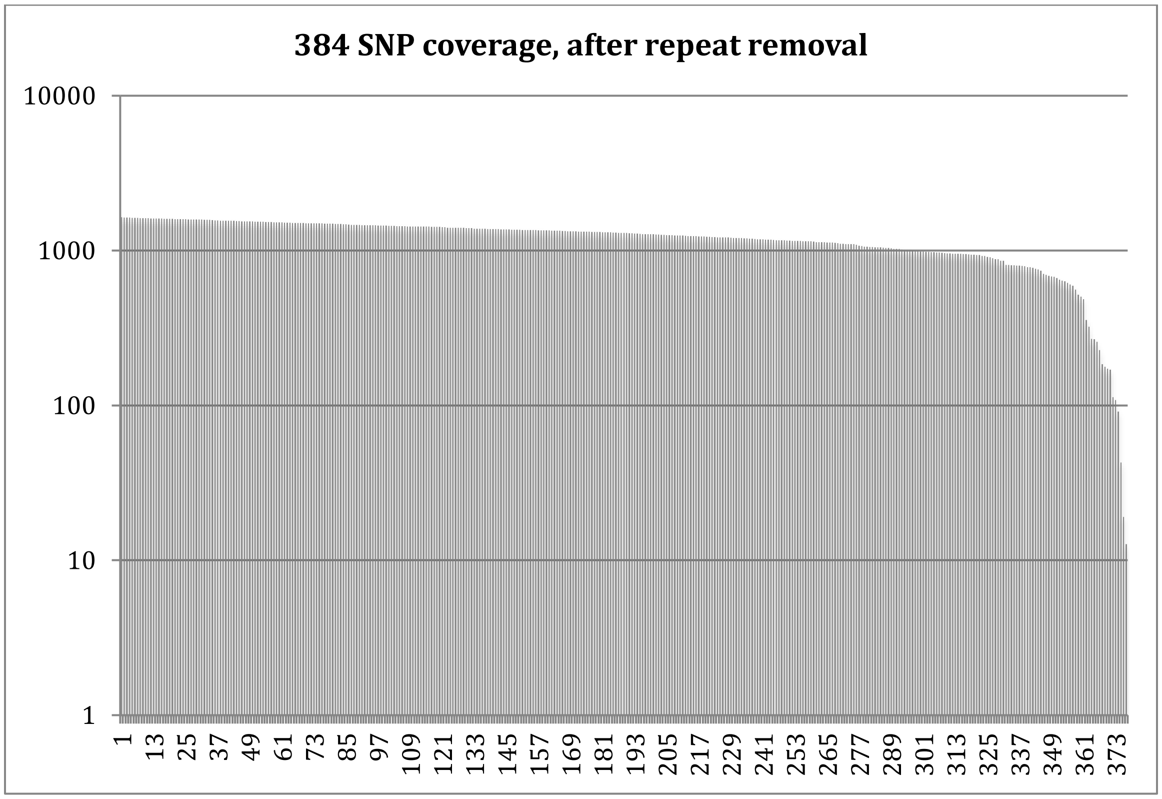

384 SNP coverage, after repeat removal
10000
1000
100
10
1
1
13
25
37
49
61
73
85
97
109
121
133
145
157
169
181
193
205
217
229
241
253
265
277
289
301
313
325
337
349
361
373

Fig. 3. Distribution of total reads assembled to 384 SNP loci a) before removal of seven repeat loci, and b) after removal and re-placement putative repeat loci. Loci falling below the threshold coverage for SNP genotyping of any samples were excluded. The Y axis is logarithmic. Plate 1 was sequenced on an Ion Torrent (458,208 aligned reads), and plate 2 was sequenced on an Ion Proton (6,198,109 aligned reads), so the Plate 2 per-locus read depth was scaled by the ratio of plate 2 to plate 1 total reads for comparison purposes.

Appendices

11 scripts (text documents):

Appendix 1: Script1_SNP_call_filter.sh

Appendix 2: Script2_generate_genotype_blocks.py

Appendix 3: Script3a_Filter1_mapping_quality.R

Appendix 4: Script3b_Filter2_excessive_coverage.R

Appendix 5: Script4a_Filter1_Remove_poor_mapping_quality.R

Appendix 6: Script4b_Filter2_Remove_excessive_coverage.R

Appendix 7: Script5_Filter3_excessive_individuals.R

Appendix 8: Script6_Filter4_Remove_rare_SNPs.R

Appendix 9: Script7_SNP_call_filter_GATK.sh

Appendix 10: Script8_generate_genotype_blocks_GATK.py

Appendix 11: Script9_Filter6_compare_SNP_datasets.R

Appendix 12: Script10_plotQC.R

Appendix 13: Script11_Filter7_Remove_HWEexcessHet.R