QTL Input Pipeline

Celine Prakash

July 26, 2019

A Pipeline to Prepare RAD-seq Data for QTL analysis

This pipeline (*QTL_input_pipeline.py*) runs the following analyses: 1. trimming of adapters with `trimmomatic` 2. merging overlapping reads with `PEAR` 3. mapping of reads with `NextGenMap` 4. variant calling with `GATK` 5. obtaining variants that are informative 6. visualising the informative variant marker positions as a heatmap of positions and genotype calls 7. detecting sites of recombination from switches in genotypes within an indivudal 8. and producing an input file for QTL analysis.

The pipeline can be found at:
`/groups/kaiserlab-ngs/05-Scripts/QTL_input_pipeline/`

1 Input for running the pipeline

1. **Input table**: The demultiplexed sequencing data files for all samples in the family have to be supplied in an input table. The table contains the sample id [col1] and fastq file(s)[col2(&3)] (with paths). See example in Figure 1.

2. **Input parameters/options**: There are a number of required parameters (mainly to specify input files and output directories) and some options for the QTL pipeline. Typing `python QTL_input_pipeline.py -h` will show the options with a description of each of them (Figure 2). However, for running of the pipeline see the next point (3.) below.

3. **Slurm `sbatch` submission script**: As running jobs on the cluster Wallace requires a `Slurm sbatch` submission, example scripts have been placed in `/groups/kaiserlab-ngs/05-Scripts/QTL_input_pipeline/` for a backcross family (Figure 3) and a F1 cross family (Figure 4). These were the submission scripts used for testing the pipeline and could be utilized as a template for modification. Note that parts of the pipeline allow for parallelization, where multiple samples can be run independently. The "`parallelize`" variable in the sbatch script determines the number of samples to run in parallel (this corresponds to the `--maxproces` parameter for the *QTL_input_pipeline.py* script). **However, the `SBATCH --ntasks` value should be modified to match this variable.**

GMH	/mnt/beegfs/celine/03_QTLpipeline/HexPor_BC_2_noReplIG_pipeline/multiplexed_reads/TCGTCTGA/AATTTCGATCC_TCGTCTGA_split_R1.fastq	/mnt/beegfs/celine/03_QTLpipeline/HexPor_BC_2_noReplIG_pipeline/multiplexed_reads/TCGTCTGA/AATTTCGATCC_TCGTCTGA_split_R2.fastq
GFP	/mnt/beegfs/celine/03_QTLpipeline/HexPor_BC_2_noReplIG_pipeline/multiplexed_reads/TCGTCTGA/ATTAGCGATCC_TCGTCTGA_split_R1.fastq	/mnt/beegfs/celine/03_QTLpipeline/HexPor_BC_2_noReplIG_pipeline/multiplexed_reads/TCGTCTGA/ATTAGCGATCC_TCGTCTGA_split_R2.fastq
MH	/mnt/beegfs/celine/03_QTLpipeline/HexPor_BC_2_noReplIG_pipeline/multiplexed_reads/TCGTCTGA/ATCGACGATCC_TCGTCTGA_split_R1.fastq	/mnt/beegfs/celine/03_QTLpipeline/HexPor_BC_2_noReplIG_pipeline/multiplexed_reads/TCGTCTGA/ATCGACGATCC_TCGTCTGA_split_R2.fastq
FHxP	/mnt/beegfs/celine/03_QTLpipeline/HexPor_BC_2_noReplIG_pipeline/multiplexed_reads/TCGTCTGA/AACCCCGATCC_TCGTCTGA_split_R1.fastq	/mnt/beegfs/celine/03_QTLpipeline/HexPor_BC_2_noReplIG_pipeline/multiplexed_reads/TCGTCTGA/AACCCCGATCC_TCGTCTGA_split_R2.fastq
1	/mnt/beegfs/celine/03_QTLpipeline/HexPor_BC_2_noReplIG_pipeline/multiplexed_reads/TCGTCTGA/ACACGTGATCC_TCGTCTGA_split_R1.fastq	/mnt/beegfs/celine/03_QTLpipeline/HexPor_BC_2_noReplIG_pipeline/multiplexed_reads/TCGTCTGA/ACACGTGATCC_TCGTCTGA_split_R2.fastq
2	/mnt/beegfs/celine/03_QTLpipeline/HexPor_BC_2_noReplIG_pipeline/multiplexed_reads/TCGTCTGA/ACTGCAGATCC_TCGTCTGA_split_R1.fastq	/mnt/beegfs/celine/03_QTLpipeline/HexPor_BC_2_noReplIG_pipeline/multiplexed_reads/TCGTCTGA/ACTGCAGATCC_TCGTCTGA_split_R2.fastq
3	/mnt/beegfs/celine/03_QTLpipeline/HexPor_BC_2_noReplIG_pipeline/multiplexed_reads/TCGTCTGA/ACGTACGATCC_TCGTCTGA_split_R1.fastq	/mnt/beegfs/celine/03_QTLpipeline/HexPor_BC_2_noReplIG_pipeline/multiplexed_reads/TCGTCTGA/ACGTACGATCC_TCGTCTGA_split_R2.fastq
4	/mnt/beegfs/celine/03_QTLpipeline/HexPor_BC_2_noReplIG_pipeline/multiplexed_reads/TCGTCTGA/AGAGTCGATCC_TCGTCTGA_split_R1.fastq	/mnt/beegfs/celine/03_QTLpipeline/HexPor_BC_2_noReplIG_pipeline/multiplexed_reads/TCGTCTGA/AGAGTCGATCC_TCGTCTGA_split_R2.fastq
5	/mnt/beegfs/celine/03_QTLpipeline/HexPor_BC_2_noReplIG_pipeline/multiplexed_reads/TCGTCTGA/AGTCATGATCC_TCGTCTGA_split_R1.fastq	/mnt/beegfs/celine/03_QTLpipeline/HexPor_BC_2_noReplIG_pipeline/multiplexed_reads/TCGTCTGA/AGTCATGATCC_TCGTCTGA_split_R2.fastq
6	/mnt/beegfs/celine/03_QTLpipeline/HexPor_BC_2_noReplIG_pipeline/multiplexed_reads/TCGTCTGA/AGCTGAGATCC_TCGTCTGA_split_R1.fastq	/mnt/beegfs/celine/03_QTLpipeline/HexPor_BC_2_noReplIG_pipeline/multiplexed_reads/TCGTCTGA/AGCTGAGATCC_TCGTCTGA_split_R2.fastq
7	/mnt/beegfs/celine/03_QTLpipeline/HexPor_BC_2_noReplIG_pipeline/multiplexed_reads/TCGTCTGA/AGGACTGATCC_TCGTCTGA_split_R1.fastq	/mnt/beegfs/celine/03_QTLpipeline/HexPor_BC_2_noReplIG_pipeline/multiplexed_reads/TCGTCTGA/AGGACTGATCC_TCGTCTGA_split_R2.fastq

Figure 1: An example input table for a paired-end dataset: with the sample ids in column 1, the R1 fastq file in column 2 and the R2 fastq file in column 3. For a single-end dataset, the fastq file is specified in column 2 only.

```
-bash-4.2$ python QTL_input_pipeline.py -h
usage: QTL_input_pipeline.py [-h] [--inputtable INPUTTABLE]
                             [--outputdirectory OUTPUTDIRECTORY]
                             [--adapterfile [ADAPTERFILE]]
                             [--mintrimmedlength [MINTRIMMEDLENGTH]]
                             [--reference REFERENCE]
                             [--minreadidentity [MINREADIDENTITY]]
                             [--minreadfraction [MINREADFRACTION]]
                             [--mergedvcffile MERGEDVCFFILE]
                             [--maxprocesses [MAXPROCESSES]]
                             [--crosstype [CROSSTYPE]]
                             [--heterozygousparent HETEROZYGOUSPARENT]
                             [--homozygousparent HOMOZYGOUSPARENT]
                             [--parents [PARENTS]]
                             [--grandparents GRANDPARENTS]

Pipeline to prepare RAD-seq data for QTL analysis:

        1. trimming of adapters with trimmomatic
        2. merging overlapping reads with PEAR
        3. mapping with NextGenMap
        4. variant calling with GATK
        5. obtaining variants that are informative
        6. converting the resulting VCF file to a matrix and producing a heatmap
        7. detecting sites of recombination
        8. producing an input file for QTL analysis.

    *Please run "module load python/2.7.13" and "module load java/x64/8u121" before using this script*

optional arguments:
  -h, --help            show this help message and exit
  --inputtable INPUTTABLE, -i INPUTTABLE
                        Input table without a header. Columns of table should be: 1. sample id  2. input fastq file (with path), [if paired-end: 3. input fastq file for read 2]
  --outputdirectory OUTPUTDIRECTORY, -o OUTPUTDIRECTORY
                        Output directory where output files should be written
  --adapterfile [ADAPTERFILE], -g [ADAPTERFILE]
                        Adapter sequences for Trimmomatic
  --mintrimmedlength [MINTRIMMEDLENGTH], -t [MINTRIMMEDLENGTH]
                        Trimmomatic minimum trimmed read length [50]
  --reference REFERENCE, -r REFERENCE
                        Reference genome file (with path)
  --minreadidentity [MINREADIDENTITY], -j [MINREADIDENTITY]
                        NextGenMap minimum read mapping identity [0.95]
  --minreadfraction [MINREADFRACTION], -s [MINREADFRACTION]
                        NextGenMap minimum read mapping fraction [1]
  --mergedvcffile MERGEDVCFFILE, -v MERGEDVCFFILE
                        Filename for merged VCF file
  --maxprocesses [MAXPROCESSES], -m [MAXPROCESSES]
                        Maximum number of processes to run in parallel [1]
  --crosstype [CROSSTYPE], -c [CROSSTYPE]
                        Cross type, either "Backcross" or "F1cross" [Backcross]
  --heterozygousparent HETEROZYGOUSPARENT, -a HETEROZYGOUSPARENT
                        If "Backcross", sample name of the heterozygous parent
  --homozygousparent HOMOZYGOUSPARENT, -b HOMOZYGOUSPARENT
                        If "Backcross", sample name of the homozygous parent
  --parents [PARENTS], -z [PARENTS]
                        If "F1cross", sample names of both F1 parents, separated by ","
  --grandparents GRANDPARENTS, -p GRANDPARENTS
```

Figure 2: The pipeline input parameter options.

2 Parts of the pipeline

A brief description of relevant parameters for the different parts of the pipeline are included in this section. Most of the parameters are not available as options for the user to modify. The only exceptions are the read mapping parameters.

2.1 `Trimmomatic` [1]

Adapter trimming in the pipeline is now done using `Trimmomatic` and is based on Nico's Fst pipeline. However, instead of the standard illumina adapter files, specific adapter sequences are provided for the ddRAD-seq libraries. An illustration of the different P1 and P2 adapters, the barcode and the multiplex index is shown in Figure 5.

Figure 3: `Sbatch` submission script for a Backcross family. Differences to the the F1 cross are highlighted in the black boxes

Figure 4: `Sbatch` submission script for an F1 cross family. Differences to the the Backcross are highlighted in the black boxes

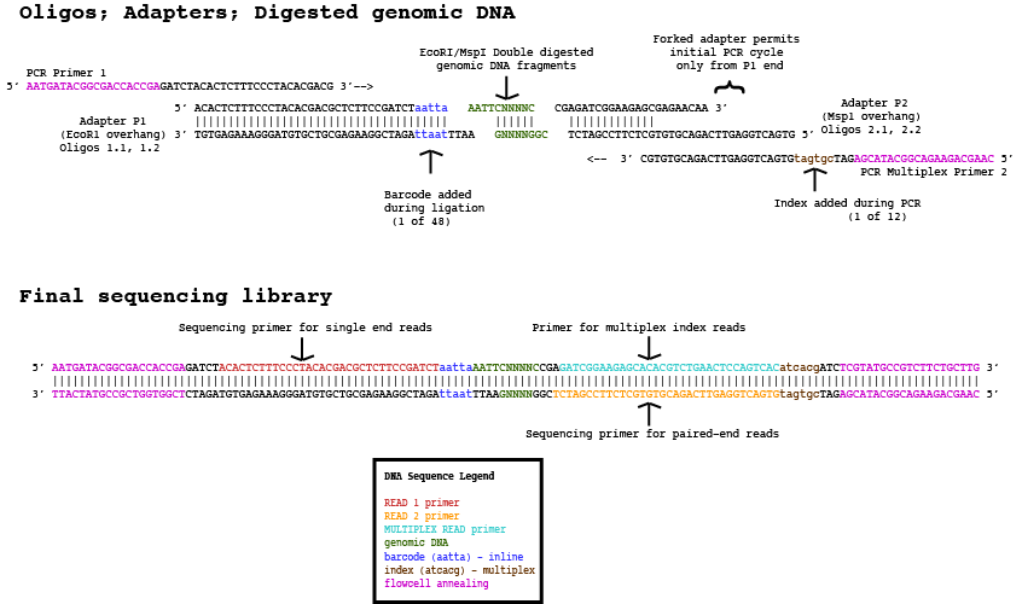

Figure 5: ddRAD adapters taken from the ddRAD protocol

2.1.1 Adapter files

All possible adapter sequences were provided by Dusica and were used to create the adapter files required by Trimmomatic.

- the paired end adapter sequence file (*QTL_pipeline_adapters_PE.fa*) contains:
  1. P1 adapters with the barcode sequence for all 60 barcodes (P1-top)
  2. the reverse complement of P1-top (i.e P1-bottom), without the restriction site overhang sequences
  3. P2 adapters for all 4 multiplex indices (P2-bottom)
  4. the reverse complement of P2-bottom (i.e. P2-top), without the restriction site overhang sequences
  5. specifically for the palindromic trimming which is meant for read through of a short fragment into the adapter sequence. To check for this, trimmomatic prepends the adapter to the start of the read and checks if the forward and reverse reads align and show a read through. Overlapping parts of the reads that match the adapter sequence are trimmed off.
     (a) the P1-top sequence for the adapter sequence that is concatenated to the start of the R1/forward read. Indicated with the name having "Prefix_*/1".
     (b) the P2-bottom sequence for the adpater sequence that is concatenated to the start of the R2/reverse read. Indicated with the name having "Prefix_*/2".

- the single end adapter sequence file (*QTL_pipeline_adapters_SE.fa*) contains:
  1. all 60 6bp-long barcode sequences
  2. the reverse complement of the barcodes
  3. the two P2 sequences Dusica used for the HexPor single end libraries
  4. the reverse complement of the two P2 sequences

2.1.2 Command parameters

Here only the changes compared to Nico's pipeline are mentioned:

- palindrome mode's min adapter length =1 and as opposed to 8 (because palindrome mode has a very low false positive rate, this can be safely reduced, even down to 1)
- Minlen = 50 (default) as opposed to 75. Specifies the minimum length of reads to be kept

2.2 PEAR [6]

This part of the pipeline is based on Nico's Fst pipeline and is only done for reads that are still paired after adapter trimming in the paired end libraries. `PEAR` assembles overlapping read pairs into a single read.

2.2.1 Command parameters

Only changes compared to Nico's pipeline are described:

- min-assembly-length = 50 (default) as opposed to 75
- cap =40 (default) as opposed to 20. This is the upper bound for the resulting quality score. (Was there a reason why this was 20 and should I switch it back?)

2.3 Interleaving unassembled read pairs [5]

Interleaving unassembled (after PEAR) paired-end reads before mapping is done to ensure that read pairs are correctly ordered.

2.4 Merging single reads into a single fastq file

The following are merged into single fastq file of single (as opposed to paired) reads

1. `PEAR` assembled paired end reads
2. `trimmomatic` unpaired reads from R1
3. `trimmomatic` unpaired reads from R2

2.5 Mapping of reads with `NextGenMap` [5]

Two options may be specified by the user when running `NextGenMap` in the QTL input pipeline:

1. minimum identity (All reads mapped with an identity lower than this threshold will be reported as unmapped)
2. min-residues, the minimum mapped read length fraction (All reads mapped with a fraction of the read length lower than the threshold will be reported as unmapped)

We thought to use 0.95 and 1 respectively, but this showed low read mapping rates in the last 3 datasets I analysed, therefore they were run with min indentity 0.9 and min read fraction 0.9. Note that with paired-end datasets, the reads that are unassembled are mapped separately from the single reads.

2.6 Sorting (and merging) and indexing of reads [3]

For sorting of paired-end datasets, the paired and single reads are sorted separately, merged and sorted again.
The following additional parameters in `samtools merge` are specified:
`-c` Combine RG tags with colliding IDs rather than adding a suffix to differentiate them.
`-p` Combine PG tags with colliding IDs rather than adding a suffix to differentiate them.

2.7 Variant calling with `GATK` [4]

This part of the pipeline is taken from Tobias' variant calling scripts and comprises:

1. `HaplotypeCaller` (by default only uses mapping q20 reads, therefore no prior filtering for maping quality of reads is required)
2. `SelectVariants`
3. `BaseRecalibrator`
4. `PrintReads`
5. `HaplotypeCaller` (on recalled mapped reads)

2.8 Merging of VCF files with `GATK` [4]

Thie step is done with `GATK`'s `GenotypeGVCFs`

2.9 No vcf filtering

In Dusica's previous analyses, she used the following filters with `vcftools` [2] on the merged vcf file:
`--maf 0.15 --max-missing 0.8 --minGQ 20`
Where: `--maf` is the minimum minor allele frequency and `--max-missing` is to exlcude

sites having missing data (0 allows sites that are completely missing and 1 indicates no missing data allowed)

However, these filters were removed from the pipeline so that the user may evaluate the quality of the data (and the maximum number of potential informative variants) before applying any filters. After evaluating the unfiltered data, the filters for the minor allele frequency and maximum missing data can be applied by rerunning the last step of the pipeline (see section 4)

The minGQ >20 (minimum genotype quality) filter did not remove any variants for 3 datasets I tested it on.

2.10 Selecting informative variants

The pipeline next runs the script *informative_variants_QTLpipeline.py*, where the vcf file is then filtered for positions that contain informative variants. An informative variant depends on user-specified cross type:

- **Backcross**: All sites that are heterozygous in the F1 parent and homozygous in the parent having a similar genotype to one of the grandparents.
- **F1 cross**: All sites where grandparents are homozygous for different alleles and at least 80% parents are heterozygous. If no parents are sequenced, only the check for grandparent's genotypes are done. Therefore one may specify `export parents=""` in the sbatch submission script

2.11 Genotype Visualisation and creation of QTL input table

In this part of the pipeline, the script *genotype_visualisation_and_classify_recombination_BACKcross_QTLpipeline.py* or *genotype_visualisation_and_classify_recombination_F1cross_QTLpipeline.py* (depending on the family's cross type) is run to visualise the informative variant marker positions as a heatmap (Figure 6 and Figure 7) of marker position genotype calls across the individuals. The script then determines sites of recombination based on switches in genotypes in the matrix of genotype calls within an individual. The genotypes are then represented as 'AA' or 'AB' (as well as 'BB' for the back cross) in a format that is suitable as an input file for QTL analysis:

- CSV format
- Missing values are recorded as 'NA'
- genotypes with alleles not found in the parents are recorded as '-'

3 Output

In the user-specified output directory, a number of subdirectories will be created for different stages of the pipeline.

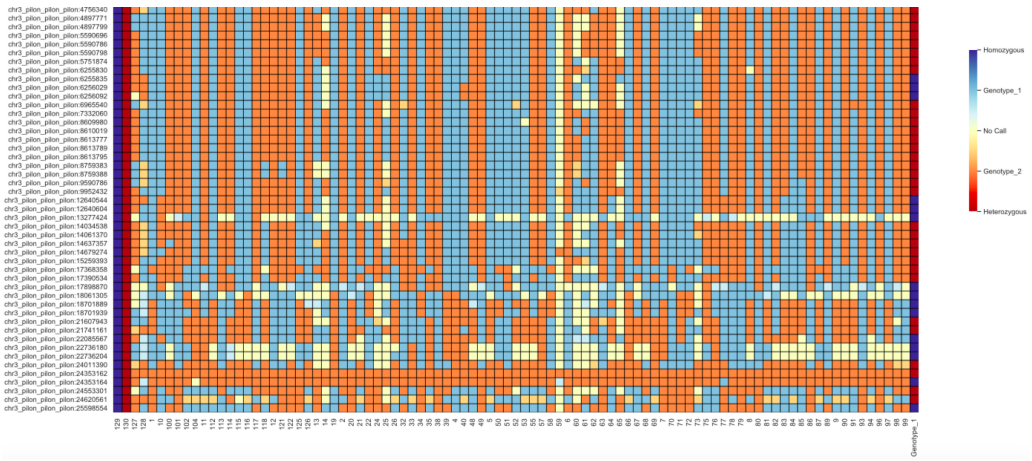

Figure 6: A subset of a heatmap of genotype calls for a Back cross family. In a Back cross heatmap, the 'homozygous' parent (with genotypes similar to a grandparent) is in the first column and the 'heterozygous' F1 parent is in the second column. Homozygous genotype calls are coloured dark blue and heterozygous calls red. As all heterozygous sites of the F1 parent are used, and it is unclear which of the homologous chromosome the allele that differs from the 'homozygous' parent belongs to, markers may switch from one homologous chromosome to the other (markers in opposition). Therefore, the genotypes of all the non-parent samples are coloured such that they reflect a conserved order of genotype calls among all the samples. Colours may be either orange or light blue. The actual genotype of the individual can be inferred by looking at the last column of the heatmap that shows the genotypes for all cells coloured light blue (i.e. Genotype_1). Orange cells would thus have the opposite genotype. Cells that are yellow are positions where there was no genotype calls. Light orange and the even lighter blue cells are positions where an unexpected allele was observed. Note that the grandparents are placed in column 3 and 4 of the heatmap

3.1 Intermediate Files

A large number of intermediate files can be found in the following sub-directories:

- `trimmed_reads`
- `assembled_reads`
- `single_reads` (the merged single reads from the paired end library described in section 2.4)
- `mapped_reads`
- `gatk_files`

  It is up to the user to delete the files. Of note the merged vcf file (with a filename specified by the user when running the pipeline) can be found in the subdirectory

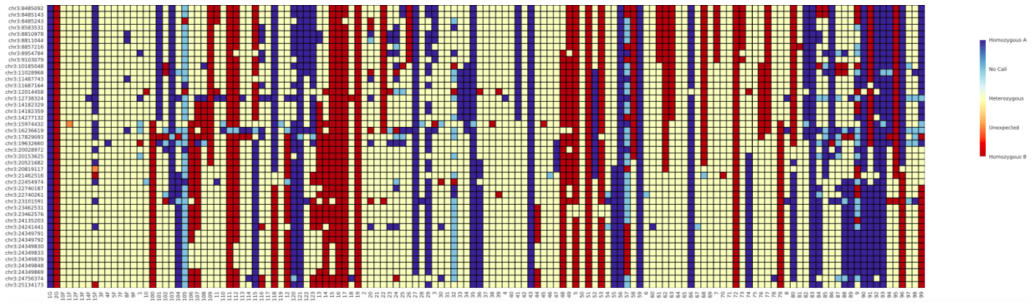

Figure 7: A subset of a heatmap of genotype calls for an F1 cross family. In an F1 cross heatmap, the grandparents are in the first 2 columns. When a genotype of a sample matches that of the grandparent, it has the same colour (either red or dark blue) as the grandparent. When it is heterozygous, it is light yellow and when the genotype is missing, it is light blue. Orange is an unexpected allele. Note that the parents, if present, are placed in column 3 onwards of the heatmap.

Pheno	Sex	ID	chr1:263363	chr1:476388	chr1:707499	chr1:968896	chr1:120361	chr1:147726	chr1:174003	chr1:2958215	chr1:295833	chr1:295994	chr1:295998	chr1:312865	chr1:339874	chr1:339876	chr1:339876	chr1:350107	chr1:350108
			1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
		1	BB	BB	BB	BB	BB	BB	BB	AB	AB	AB	AB	AB	AB	AB	AB	AB	AB
		10	BB	BB	BB	BB	BB	BB	AB	AB	AB	AB	AB	AB	AB	AB	AB	AB	AB
		100	AB	AB	AB	AB	AB	AB	AB	BB	BB	AB	AB	AB	AB	AB	AB	AB	AB
		101	AB	AB	BB	AB	AB	AB	AB	BB	BB	BB	BB	BB	BB	BB	BB	BB	BB
		102	AA	AA	AA	NA	AA	AA	AA	AA	AA	AA	AA	AA	AA	AA	AA	AA	AA
		103	AA	AA	AA	AA	AA	AA	BB	BB	BB	AB	AB	BB	AB	AB	AB	AB	AB
		104	BB	BB	NA	AB	AB	AB	AB	AB	AB	AB	AB	BB	AB	AB	AB	AB	AB
		105	AB	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA
		106	AA	AA	AA	AA	AA	AA	AA	AA	AA	AA	AA	AA	AA	AB	AB	AA	AA
		107	BB	BB	BB	AB	AB	AB	AB	AB	AB	AB	AB	AB	AB	AB	AB	AB	AB
		108	BB	BB	NA	BB	BB	BB	BB	BB	BB	BB	BB	BB	BB	BB	BB	BB	BB
		109	AB	AB	AB	AB	AB	AB	AB	AB	AB	AB	AB	AB	AB	AB	AB	AB	AB
		11	AB	AB	AA	AA	NA	AA	AA	AA	AA	AA	AA	AA	AA	AB	AB	AA	AA

Figure 8: A subset of the QTL input table produced at the end of the pipeline, to be used as an input for QTL analysis. Marker positions are specified in the columns and individuals are specified in the rows. Genotypes are represented as 'AA' or 'AB' (as well as 'BB' for the back cross)

`gatk_files` and might be worth keeping. Storing the file in a compressed version can be done with

```
bgzip -c <vcffile> > <vcffile>.gz
tabix -p vcf <vcffile>.gz
```

3.2 Final Files

The final result files can be found in the following sub-directory:

- `informative_variants`

  This subdirectory contains:

  1. pdf plots per chrom visualising the genotypes across the samples (also the matrices that were used to create these plots '.txt' files)
  2. a QTL input table (csv format)

  In the output directory, the following files are also produced:

  - A file (*QTLinput_command.txt*) that contains a command that was used to generate the pdf files and csv file in (1. and 2.). This command can be modified by the user to their desired filters and then rerun to produce the filtered versions of the output files.
  - A file (*pdfunite_command*.txt*) that contains a command for '`pdfunite`' that can be used to merge the individual pdfs per chrom into a single pdf.

4 Re-running the QTL input command

By modifying the file (*QTLinput_command.txt*) and copying and entering the commands into the command line, a re-run of genotype visualisation and QTL input table can be done. The following filtering options can be specified (which is applied to each marker position):

- `--depthfilter` minimum depth
- `--fractionindividuals` minimum fraction of individuals meeting the minimum depth requirement
- `--mendelianfraction` minimum fraction for mendalian inheritance: the distribution of either heterozygous or homozygous individuals should be at least a cerain fraction of all individuals
- `--maxdiffmissing` max samples that can have an unexpected allele are missing genotype calls
- `--mergedistance` distance between marker positions for collapsing markers to a consensus genotype (these are assumed to be from the same RAD marker)
- `--maxrecombinationindividuals` max number of samples expected to have recombination (marker positions with too many individuals showing recombination are removed)
- `--minmarkersforrecombinationregion` min number of markers between 2 recombination events to be considered as a true recombinations and not miscalls.
- `--excludesamples` sample ids of individuals to be excluded separated by ","

References

[1] Anthony M Bolger, Marc Lohse, and Bjoern Usadel. Trimmomatic: a flexible trimmer for illumina sequence data. *Bioinformatics*, 30(15):2114–2120, 2014.

[2] Petr Danecek, Adam Auton, Goncalo Abecasis, Cornelis A Albers, Eric Banks, Mark A DePristo, Robert E Handsaker, Gerton Lunter, Gabor T Marth, Stephen T Sherry, et al. The variant call format and vcftools. *Bioinformatics*, 27(15):2156–2158, 2011.

[3] Heng Li, Bob Handsaker, Alec Wysoker, Tim Fennell, Jue Ruan, Nils Homer, Gabor Marth, Goncalo Abecasis, and Richard Durbin. The sequence alignment/map format and samtools. *Bioinformatics*, 25(16):2078–2079, 2009.

[4] Aaron McKenna, Matthew Hanna, Eric Banks, Andrey Sivachenko, Kristian Cibulskis, Andrew Kernytsky, Kiran Garimella, David Altshuler, Stacey Gabriel, Mark Daly, et al. The genome analysis toolkit: a mapreduce framework for analyzing next-generation dna sequencing data. *Genome research*, 20(9):1297–1303, 2010.

[5] Fritz J Sedlazeck, Philipp Rescheneder, and Arndt Von Haeseler. Nextgenmap: fast and accurate read mapping in highly polymorphic genomes. *Bioinformatics*, 29(21):2790–2791, 2013.

[6] Jiajie Zhang, Kassian Kobert, Tomáš Flouri, and Alexandros Stamatakis. Pear: a fast and accurate illumina paired-end read merger. *Bioinformatics*, 30(5):614–620, 2013.