Mutational Genomics

Dan MacLean

2017-05-16

Contents

Preface **7**
A fork is not a hairbrush - it just has some similar properties 7
So finding the function of a gene isn't straightforward 8
We need something smarter - mutational genomics is this smarter thing 8
Our objective 8

1 Mutagenesis **11**
1.1 Learning Objectives 11
1.2 Mutagenesis with EMS 11
1.3 Getting a population homozygous for EMS induced mutations 11
1.4 Genetic screens 12
1.5 Recombination during crossing causes changes in SNP density away from the site of mutation 13
1.6 Section Quiz 13

2 PreProcessing Data **15**
2.1 Learning Objectives 15
2.2 Fastq 15
2.3 FastQC 16
2.4 Exercises 17

3 Aligning Reads To A Reference **19**
3.1 Learning Objectives 19
3.2 Mapping and Alignment 19
3.3 Exercises 21

4 Finding SNPs With An Alignment **23**
4.1 Learning Objectives 23
4.2 MPileup 23
4.3 Exercises 25

5 Visualising SNPs To Find Candidates **27**
5.1 Learning Objectives 27
5.2 Annotation 27
5.3 CandiSNP 28

5.4 Density plots 28
5.5 Centromeres 29
5.6 SNP Deletion - Fewer are better 29
5.7 Exercises 30

List of Figures

1 Ariel thinks the fork is a brush, it does look like one... 7
2 One of these is the key to the bathroom. I hope that you're not desperate! 8
3 One of these mice is not like the other mice - it has a phenotype change 9

1.1 A plant mutagenesis scheme from Page and Grossniklaus (2002). 12
1.2 One of these plants is affected in the pathways that control flowering time - the right hand plant flowers early. Source: Detlef Weigel 13
1.3 Consider crossing two chromosomes. Recombination causes the parts furthest from the selected mutation to lose the homozygous mutations as a function of distance from the selected mutation. Source: Ryan Austin[1] 14

2.1 FastQC Summary plot. Along the x-axis the plot shows the position in the read and for each position in the reads it shows a box-plot of all the quality scores at that position. . . . 16
2.2 The first few bases here are significantly enriched, this can be due to sequence adapters (if they were used) but if not, then the sequence is likely not good, even if the quality scores are fine. 17

5.1 Screenshot of the CandiSNP tool. Spots represent SNPs (height on the y axis shows major allele frequency). 28
5.2 CandiSNP after filtering. The region of the high red spot density is the recombinant region 29
5.3 Density plot of homozygous, heterozygous SNP density and the ratio of hom/het SNPS in sliding windows 29

---

[1] http://bar.utoronto.ca/ngm/description.html

Preface

A fork is not a hairbrush - it just has some similar properties

Genomics has come a long way. We can now sequence genomes quickly and to a reasonable degree of accuracy such that with sequence based approaches we can create in a high-throughput manner an inventory of sub-regions in a genome that we think are genes. We know the functions (or some of the functions) of lots of genes and we can infer functions of newly discovered genes by comparison of sequence or structure, basically by seeing whether our new thing looks like something else.

But these methods are actually only PREDICTIONS of function. Looking a bit like something else is only a clue to what something does. It frequently fails us. We make the same mistake as Ariel in Figure 1.

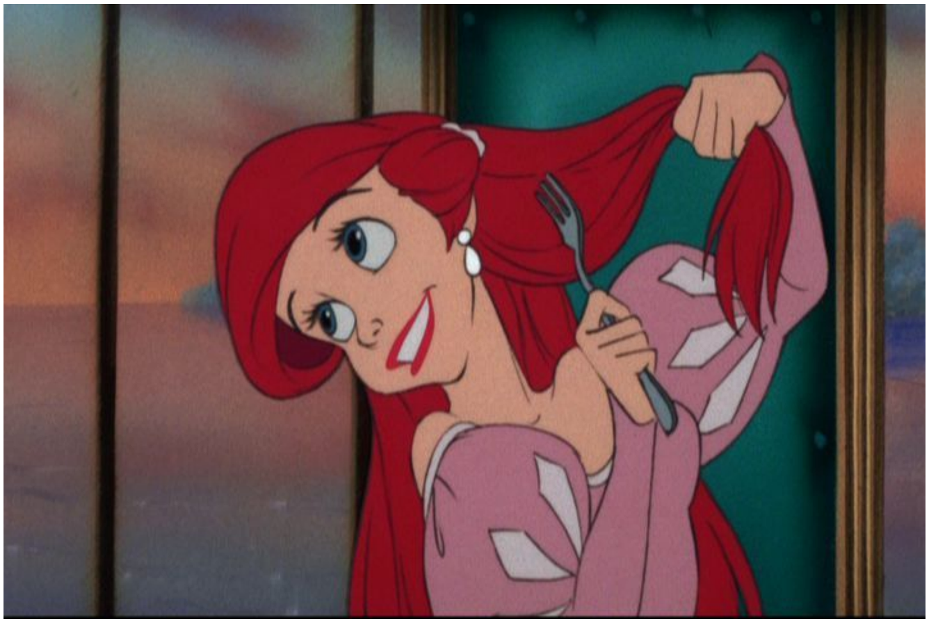


Figure 1: Ariel thinks the fork is a brush, it does look like one...

Figure 2: One of these is the key to the bathroom. I hope that you're not desperate!

So finding the function of a gene isn't straightforward

What makes things worse is that often we don't start with the gene itself. We start with some biological process and want to find genes with a role in that process. Because our functional predictions aren't perfect we can't collect all the genes and start trying them out one-by-one. It could take a while ... Figure 2.

We need something smarter - mutational genomics is this smarter thing

With mutational genomics we deal initially with the effect of the gene on the whole organism (Figure 3). By performing mutagenesis on our favourite organism then carrying out a screen that selects individuals that have changed in the phenotype we are interested in, we have our first foothold. We can study those individuals and apply the principles of genetics, use modern high-throughput sequencing and bioinformatics tools to identify the gene causing that phenotype change (or at least ones involved in the process we have messed up).

Our objective

This will be the focus of this workshop - how to go from samples identified in a genetic screen to a short-list of candidate genes using Galaxy tools and software.

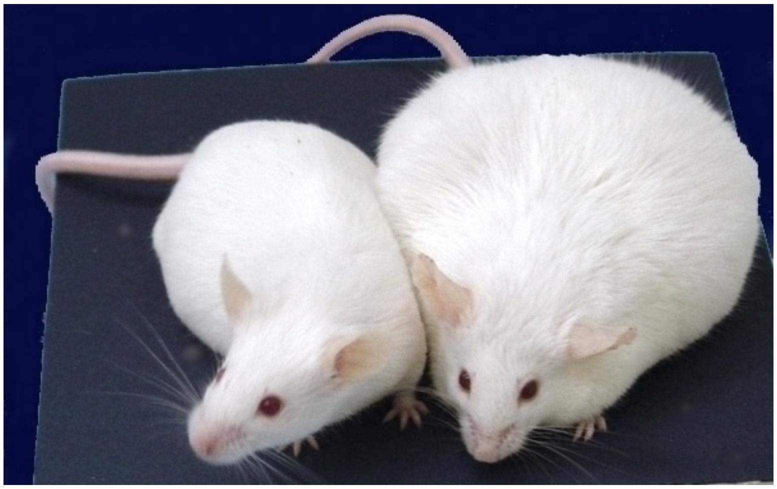


Figure 3: One of these mice is not like the other mice - it has a phenotype change

Chapter 1

Mutagenesis

1.1 Learning Objectives

- Mutagens make changes in a genome
- EMS creates Single Nucleotide Polymorphisms of C->T or G->A Transition (mostly)
- Careful crossing gives us homozygous mutant lines
- Genetic screens select lines with changes related to our interest

1.2 Mutagenesis with EMS

The first step is to mutagenise a population of organisms, or cells or similar. Mutagenesis is basically causing damage to DNA. Lot's of things can do this, Wikipedia has a great page on mutagens[1]. In plant genetics, the mutagen we typically use for changing single nucleotides in DNA, things we call point mutations or Single Nucleotide Polymorphisms (SNPs), is EMS - Ethyl methanesulfonate[2]. EMS will predominantly make C's change to T's and G's change to A's. These mutations are distributed fairly uniformly throughout the genome.

In practice we take a load of plant seeds and soak them in a solution of EMS. The EMS soaks in and damages the plant's DNA.

This damages the DNA in **some** but **not all** of the cells in the seed.

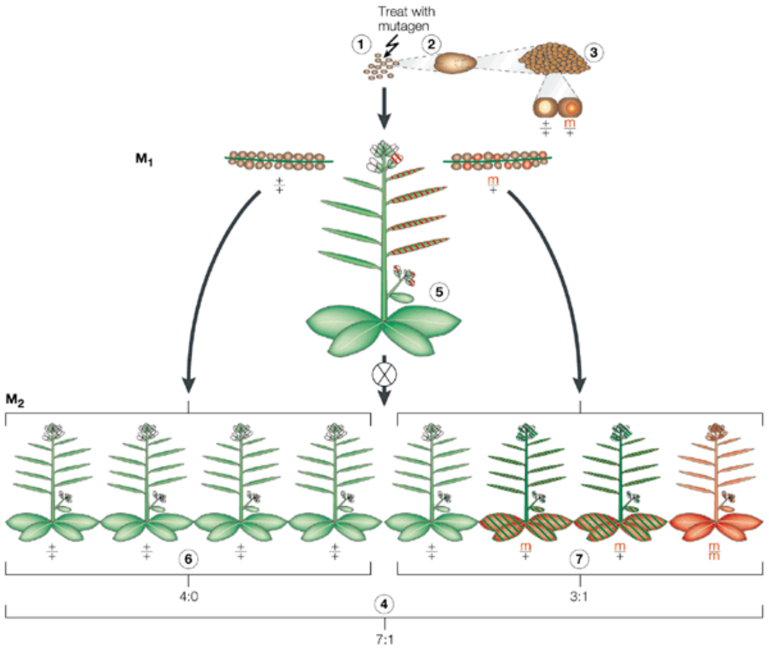
1.3 Getting a population homozygous for EMS induced mutations

We grow up the seed (let's call the plants that grow up the M1 generation) and the cells in those M1 plants that descended from the mutagenised cells carry the mutations. Sometimes these will be cells in the germline - ones that beget seeds. The progeny of the M1 plants, that grow from mutation carrying seeds

---

[1] https://en.wikipedia.org/wiki/Mutagen

[2] https://en.wikipedia.org/wiki/Ethyl_methanesulfonate



Nature Reviews | Genetics

Figure 1.1: A plant mutagenesis scheme from Page and Grossniklaus (2002).

(let's call these progeny the M2) will all grow up with the mutation **in every cell** and eventually, by identifying the progeny plants carefully, we can get a population of plants which are homozygous for all the EMS SNP mutations we induced.

This same principle is true for any organism. Whatever you want to do mutational genomics with, you will need to:

1. mutate
2. select
3. cross
4. screen

1.4 Genetic screens

Once we have a mutagenised population we can start to select the individuals in that population that show some change in the phenotype of interest, say flowering time. We can use further crosses to bring in extra variation or use the natural variation - both of which are heterozygous - while keeping the homozygous mutation(s) that is (are) causing the phenotype by constantly selecting for offspring that show the phenotype of interest everytime we carry out crosses.

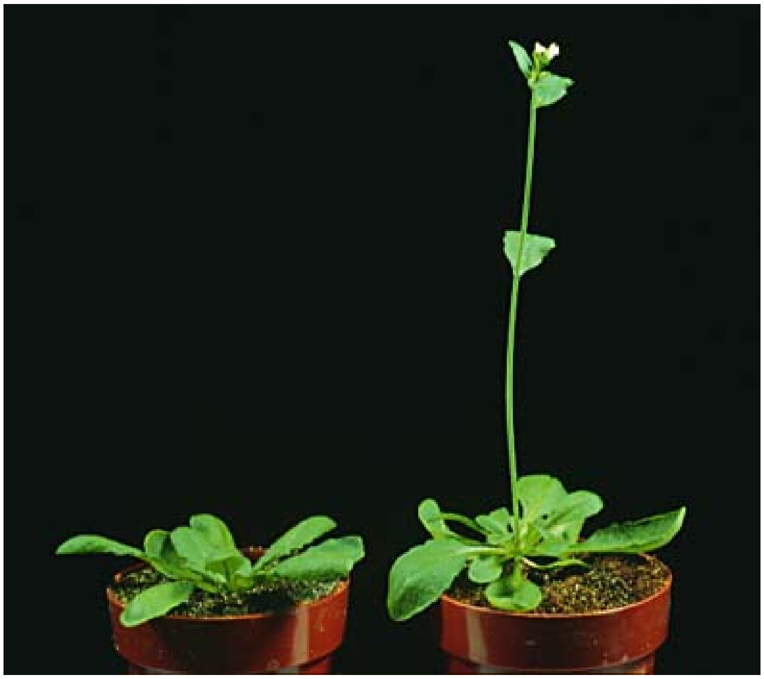


Figure 1.2: One of these plants is affected in the pathways that control flowering time - the right hand plant flowers early. Source: Detlef Weigel

1.5 Recombination during crossing causes changes in SNP density away from the site of mutation

All these crosses cause recombination in the chromosomes that bring in homologous parts from the line being used to cross. This happens at a greater frequency away from the mutation that we are selecting for such that regions further from the selected mutation carry fewer and fewer of the homozygous mutations.

This statistical difference, a region of high homozygous SNPs around the mutation is what we will use to identify the SNPs that cause our phenotype of interest.

1.6 Section Quiz

Please now complete the section quiz at this link[5]

---

[5] https://goo.gl/forms/RLGLndlcEcoV8c742

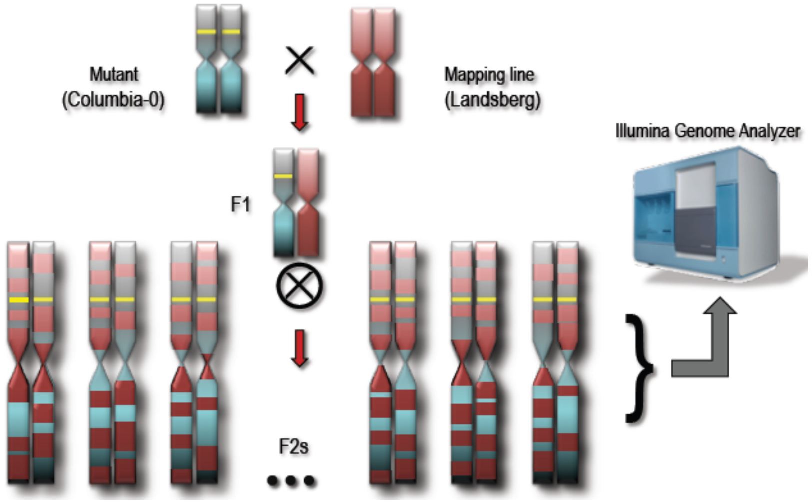

Mutant
(Columbia-0)
Mapping line
(Landsberg)
Illumina Genome Analyzer
F1
F2s

Figure 1.3: Consider crossing two chromosomes. Recombination causes the parts furthest from the selected mutation to lose the homozygous mutations as a function of distance from the selected mutation. Source: Ryan Austin[4]

Chapter 2

PreProcessing Data

2.1 Learning Objectives

- Understanding Fastq
- Inspecting and interpreting the quality of sequence data with FASTQC
- Cleaning out sequence data with trimmomatic.

2.2 Fastq

Fastq[1] is a typical sequence format generated by HTS machines. It contains four sections, a sequence ID, the sequence, the ID again and a messy looking quality string made up of characters, each of which represents the quality of the base above it. Here's an example:

```
@SEQ_ID
GATTTGGGGTTCAAAGCAGTATCGATCAAATAGTAAATCCATTTGTTCAACTCACAGTTT
+SEQ_ID
!''*((((***+))%%%++)(%%%%).1***-+*''))**55CCF>>>>>>CCCCCCC65
```

Each of the weird characters represents a number according to the `ASCII`[2] look up table, where numbers are linked to characters, so `!` means 33 and `"` means 34. These numbers are generally `Phred`[3] scores, which encode the likelihood of the base being wrong on a log scale.

We can use this quality information to assess how well our sequencing went. Along with sequence quality information we should also assess the composition of the sequence data. A program called FastQC (Andrews, 2017) is useful for this.

---

[1] https://en.wikipedia.org/wiki/FASTQ_format

[2] https://en.wikipedia.org/wiki/ASCII

[3] https://en.wikipedia.org/wiki/Phred_quality_score

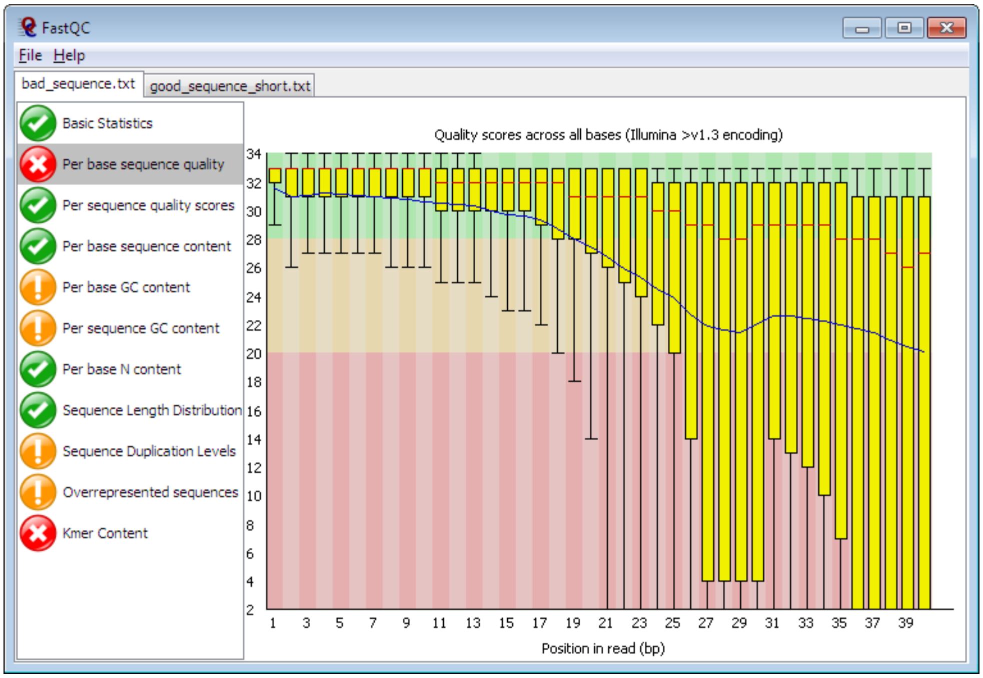

FastQC
File Help
bad_sequence.txt
good_sequence_short.txt
Basic Statistics
Per base sequence quality
Per sequence quality scores
Per base sequence content
Per base GC content
Per sequence GC content
Per base N content
Sequence Length Distribution
Sequence Duplication Levels
Overrepresented sequences
Kmer Content
Quality scores across all bases (Illumina >v1.3 encoding)
34
32
30
28
26
24
22
20
18
16
14
12
10
8
6
4
2
1 3 5 7 9 11 13 15 17 19 21 23 25 27 29 31 33 35 37 39
Position in read (bp)

Figure 2.1: FastQC Summary plot. Along the x-axis the plot shows the position in the read and for each position in the reads it shows a box-plot of all the quality scores at that position.

2.2.1 Quality score encoding

Different sequencers use slightly different variations on the Phred score, this is usually called the quality encoding. Older Illumina pipelines encoded a score from -5 to 62 using ASCII characters 59 to 126, but nowadays most use Sanger encoding, which encodes a score from 0 to 93 using ASCII 33 to 126.

Because of this discrepancy, it is necessary to sometimes be explicit about the sequence encoding in Galaxy. We do this by setting the data attributes of data files

2.3 FastQC

FastQC presents a range of plots and summary statistics, you need to provide it with Fastq data.

A typical output like that in Figure 2.1 shows the `per base sequence quality`.

The box plots to the left have much higher and tightly grouped quality scores than those on the right. This is typical of Illumina machine sequence, the quality decreases the further you get along the read. As you can infer from the red region of the plot background, Phred scores less than 20 are generally not trusted.

We may (or may not) decide that we need to get rid of the lower quality sequence.

At the individual sequence read level, we can discard entire sequences if part is too poor or trim the read leaving the good part alone. One system for doing this is (Bolger et al., 2014) which can perform a variety of trimming operations on sequence reads. It can remove parts of reads from the left or right sides up to quality thresholds - it uses a sliding window average, rather than just a harsh cut-off.

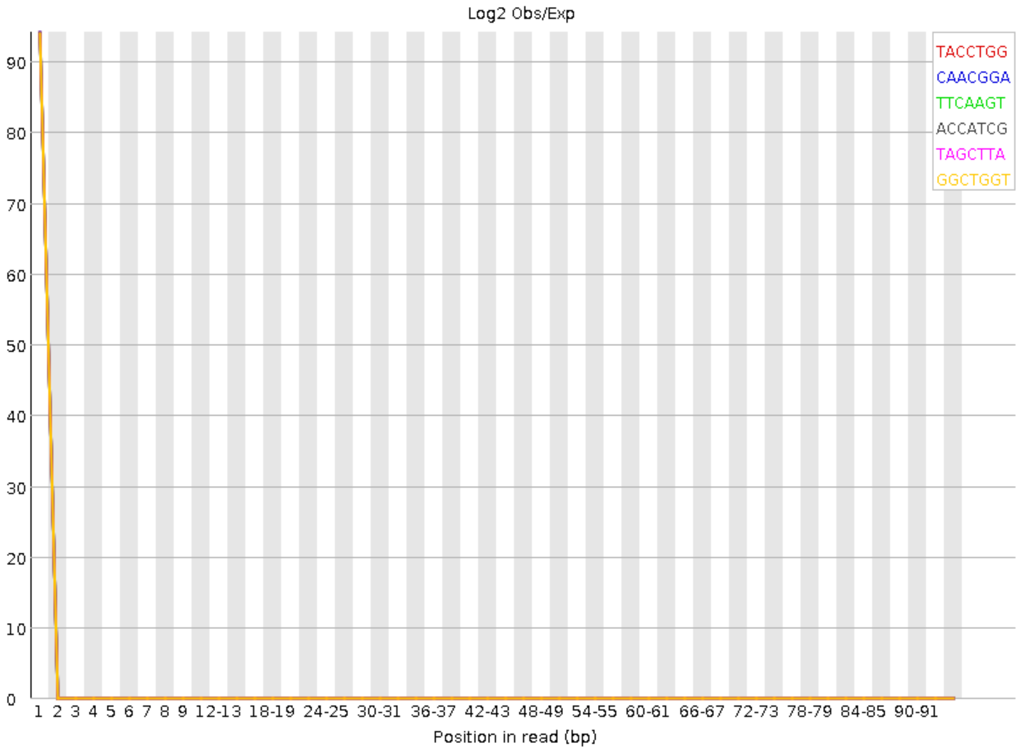


Figure 2.2: The first few bases here are significantly enriched, this can be due to sequence adapters (if they were used) but if not, then the sequence is likely not good, even if the quality scores are fine.

2.3.1 Sometimes we shouldn't trim

If we do carry out trimming, then we may end up with lots of reads of different lengths - this can be a problem for some aligners and downstream tools, so sometimes trimming isn't the best strategy, we have to make context dependent decisions.

At the sequence sample level (e.g, the read file level), we may discover that our read set is not good. Reports from FastQC like the *k*-mer content plot (Figure 2.2) can show sequence problems, in this graph there is an over-representation of particular *k*-mers at the start of the sequence that shouldn't be there.

2.4 Exercises

Your task now is to load up Galaxy and run some reads through quality control and trimming, prior to downstream use.

2.4.1 Power Up The VM

1. Start `VirtualBox` by double-clicking it's application icon.
2. Use `File` .. `Import Appliance` and select the `??????` VM file.

2.4.2 Start Galaxy

Galaxy should appear ready and waiting in the Chromium browser on the desktop in the VM. The bookmarks bar in this browser has all the links you need for this workshop.

2.4.3 Run FastQC

Use the reads in the `Pre-processing` data library. You will find the `FastQC` tool in the tool list under `HTS QC`. The reads are single ended from mutagenised *Arabidopsis thaliana*. They are Illumina Whole Genome Shotgun reads[4] so don't have any multiplex adapters and the plants are grown in sterile culture so we aren't expecting contamination.

1. How many reads are you using?
2. What sort of output files do you get from FastQC?
3. What should you do with these files? Do they represent a scientific control that could be published?

2.4.4 Interpret Sequence Quality

1. Is there any evidence of contamination? Which report tells you?
2. If there is, which sequence is contaminating?
3. Which of the four files is the best?

2.4.5 Clean Up Poor Quality Sequence

Use the `Trimmomatic` tool in `HTS QC`. Select an appropriate `Average quality required`

1. Find a trimming strategy to get rid of contamination you observed in section `2.1`?
2. Which trimming strategy improves the low quality reads
3. Did it make any difference?
4. How could you filter on size? If you needed to pass only good quality, full length sequences to the next step?

---

[4] https://en.wikipedia.org/wiki/Shotgun_sequencing

Chapter 3

Aligning Reads To A Reference

3.1 Learning Objectives

- Understanding Alignment
- Know how to run the correct BWA alignment
- Understand the SAM/BAM format and relationship

  Culture Clash

  If you're a bioinformatician - you may be wondering why I'm not using the words 'mapping' reads here. Well, the geneticists in the crowd have a much older technique called mapping[1] that does something completely different. I have honestly had conversations where it has been insisted that I can't possibly map a read. Don't suffer as I did.

  If you're a geneticist - bioinformaticians claim they can do amazing things with reads, mapping them down to the single read level! Of course they usually just mean they're aligning them to whatever reference sequence they have.

3.2 Mapping and Alignment

Mapping is the process of finding the position on a genome that a read best matches to, so is likely to have come from. Alignment is the optimal alignment of the sequence of the read with the reference sequence, such that gaps and errors are allowed and small variations can be found.

3.2.1 BWA

There are many tools for aligning reads (too many, probably). One of the best general ones is BWA (Li and Durbin, 2010). BWA is a high-throughput sequence aligner used to align relatively short sequences to a reference genome. It uses the Burrows-Wheeler Transform reduce the amount of memory needed to align reads by creating a compressed index of the reference sequence. It contains two algorithms for alignment, one `aln` for short reads up to around 200 bp with low error rate and another `mem` for longer reads. Both are very fast and accurate.

---

[1] https://en.wikipedia.org/wiki/Gene_mapping

BWA is complicated and there are lots of options to set. The crucial things we are going to need are a reference genome, and a set of reads.

Often, a sequencing strategy will use paired-end reads[2], where we know the distance between two reads and we can set that as a parameter. Usually the reads from a paired strategy come in two files, one of the pair in a 'left' file, and the other in a 'right' file.

BWA has a lot of options, here's some important ones:

1. `Maximum edit distance` - is the maximum number of nucleotides in a read that can mis-match with the reference and the read still be aligned.
2. `Maximum number of gap opens`- refers to the amount of insertions that can occur across a read.
3. `Disallow insertion/deletion within some bp towards the end` - allows for the fact that sequence quality deteriorates towards the end of a sequence and the user might not want to trust indels in the last few bases of a read.
4. `Maximum insert size for a read pair to be considered as being mapped properly` - The insert size of reads is the distance between the outer ends of the two paired-end reads.

3.2.2 SAM Format

The output format from BWA and most other aligners is Sequence Alignment Map[3] (SAM) format (Li et al., 2009). The SAM format describes the alignment of sequenced reads to a reference sequence. It stores all the alignment information generated by BWA in a simple and compact format. It provides information about the position of the read in relation to the reference genome, the number and position of nucleotides that match to the genome and the position of indels. SAM files are often analysed in packages like SAMTools (Li et al., 2009) and Picard (Picard[4])

Here's the top of a SAM file:

```
@HD VN:1.3  SO:coordinate
@SQ SN:chloroplast  LN:154478
@PG ID:bwa  PN:bwa  VN:0.7.10-r876-dirty    CL:bwa mem -t 1 -v 1
chloroplast-1781    99  chloroplast 54  60  250M    =   374 570 TTA A?? NM:i:0  MD:Z:250    AS:i:250    XS:i
chloroplast-757 163 chloroplast 66  60  250M    =   459 643 GCT A5? NM:i:6  MD:Z:46A20T65T56A23A2T32    AS:i
chloroplast-1781    147 chloroplast 374 60  250M    =   54  -570    ACT G:E NM:i:6  MD:Z:9T42G1G2G7C42A141
chloroplast-703 163 chloroplast 437 60  250M    =   794 607 AGC ??? NM:i:8  MD:Z:68T2C20C68T16G9A30A8G21
```

1. The first few lines start with stuff like `@SQ SN` which described thigs like program parameters followed by the name of the sequences in the reference file (`chloroplast`) and the length of the sequence (154478).
2. Every line after that corresponds to each read that BWA handled.
3. Each line starts with the name of the read and has a number of columns of data after it.

---

[2] https://www.illumina.com/technology/next-generation-sequencing/paired-end-sequencing_assay.html

[3] https://en.wikipedia.org/wiki/SAM_(file_format)

[4] https://broadinstitute.github.io/picard/

4. In the 3rd column we can see which chromosome our read has been mapped to and the position of the first mapped base in column 4. An unmapped read will have a * in column 3 and zero in column 4.
5. The 5th column is the mapping quality, a score quantifying the probability that a read is misplaced.
6. The 6th column is the CIGAR string and consists of numbers followed by an upper-case letter (the operator), which describes the alignment.

As an example, a CIGAR string of `36M3D40M` means 36 matching nucleotides, followed by 3 deletions and ending in 40 matches.

A BAM file is a highly compressed, indexed binary version of SAM and through a library of different tools, allows fast, random access to the alignment. Some Galaxy tools will automatically convert the output from aligners to BAM format for you - (BWA does this!).

3.3 Exercises

3.3.1 Align Paired-End Reads To A Reference Genome

Use the two sets of paired reads in the `Alignment` shared data library and the `ATH1_chloroplast` reference genome sequence, to carry out a HTS alignment. Again these are sequences from the chloroplast genome of the model plant *Arabidopsis* . One set of reads, `MS`, are from an Illumina MiSeq machine, are 250 nt long and have a fragment length of 650 nt. The others, `GA2` are from an Illumina GAII machine, are 75 nt long and have a fragment length of 350.

1. Which algorithm should you use for each set of reads?
2. Align each of these with the BWA program in the `HTS Alignment` tools section. Choose an appropriate algorithm for each sequence set.
3. Pick parameters to make the two alignments as accurate and equivalent as possible? Which should differ? Which should be the same?
4. Check the results with the SAMtools `idxstats` tool and other alignment stats tools like `Flagstat` and `Stats` in `HTS SAMtools`. How do the results relate to what you know about the sequencing strategy? (What are the calculated insert sizes, what is the coverage, how many reads do you keep?)

3.3.2 Merge Alignments Into One BAM

After alignment of two read sets from different sequencing strategies, you might want to merge all of them into one BAM file so that you can work on them as one. This isn't just more convenient, it lets you make use of the extra information from combining the reads into one coverage pileup when calling SNPs or visualising the alignment.

Merging is a multi-stage process that can be done with Picard[5], the steps in Picard look like:

1. Remove duplicate reads
2. Sort BAMs individually

---

[5] https://broadinstitute.github.io/picard/

3. Merge BAMs

Picard tools are available in the `Picard` tool section. Try merging the BAM files.

3.3.3 Alignment Quality

The `HTS SAMtools` tool `BAM-to-SAM` will allow you to turn the binary BAM file into a SAM file you can read.

1. How good do the individual alignments look overall? Can you tell from the output? Is it useful to look at single alignments one by one?
2. Does the SAM Specification[6] give you any further information from which you can judge the alignments?
3. In what cases might the individual alignments be useful?

---

[6] https://samtools.github.io/hts-specs/SAMv1.pdf

Chapter 4

Finding SNPs With An Alignment

4.1 Learning Objectives

- Understanding Pileups and VCFs
- Calling reliable SNPs
- Annotating SNPs with SNPEff

Many different programs can be used to call SNPs, including SAMTools (Li et al., 2009), Picard, GATK (De Summa et al., 2017) and VarScan (Koboldt et al., 2009). Some of these programs use the BAM alignment file directly, some use an MPileup file. MPileups and VCF files are both used to represent SNPs and associated scores.

4.2 MPileup

The MPileup format is a per-line base-by-base representation of the alignment. It provides a much more coherent way of viewing the alignment than the SAM file.

Here's a sample:

```
seq1 272 T 24  ,.$.....,,.,.,...,,,.,..^+. <<<+;<<<<<<<<<<<=<;<;7<&
seq1 273 T 23  ,.....,,.,.,...,,,.,..A <<<;<<<<<<<<<3<=<<<;<<+
seq1 274 T 23  ,.$....,,.,.,...,,,.,...    7<7;<;<<<<<<<<<=<;<;<<6
seq1 275 A 23  ,$....,,.,.,...,,,.,...^l.  <+;9*<<<<<<<<<=<<:;<<<<
seq1 276 G 22  ...T,,.,.,...,,,.,....  33;+<<7=7<<7<&<<1;<<6<
seq1 277 T 22  ....,,.,.,.C.,,,.,..G.  +7<;<<<<<<<&<=<<:;<<&<
seq1 278 G 23  ....,,.,.,...,,,.,....^k.   %38*<<;<7<<7<=<<<;<<<<<
seq1 279 C 23  A..T,,.,.,...,,,.,..... ;75&<<<<<<<<<=<<<9<<:<<
```

Each line represents a single nucleotide in the reference. The first three columns represent the reference name, position on the reference and the reference nucleotide.

The next three columns are about the bases piled-up over that position, so are total read depth, read bases, and base qualities.

At the read base column,

- a dot `.` = a match to the reference base on the forward
- a comma `,` = match on the reverse strand
- any of `ACGTN` = a mismatch on the forward strand
- any of `acgtn` = mismatch on the reverse strand

4.2.1 Different Flavours of PileUp

Pileup format has been extended at various times, so the exact format you get can vary a little, the description above is the core of them all and the extensions usually provide some further quality information. Recent versions of SAMtools add quite a few columns to this description.

You can see more here http://samtools.sourceforge.net/pileup.shtml

`SAMtools` is usually used to generate an MPileup, the `mpileup` command can do this. More recent versions of `SAMtools mpileup` and most other SNP callers can also generate another type of SNP describing file, a VCF Variant Call Format[1] file.

The related VCF file takes a slightly different approach and looks like this:

```
##FORMAT=<ID=DP,Number=1,Type=Integer,Description="Read Depth">
##FORMAT=<ID=HQ,Number=2,Type=Integer,Description="Haplotype Quality">
#CHROM POS ID REF ALT QUAL FILTER INFO FORMAT NA00001 NA00002 NA00003
20 14370 rs6054257 G A 29 PASS NS=3;DP=14;AF=0.5;DB;H2 GT:GQ:DP:HQ 0|0:48:1:51,51 1|0:48:8:51,51 1/1:43:5:.,
20 17330 . T A 3 q10 NS=3;DP=11;AF=0.017 GT:GQ:DP:HQ 0|0:49:3:58,50 0|1:3:5:65,3 0/0:41:3
20 1110696 rs6040355 A G,T 67 PASS NS=2;DP=10;AF=0.333,0.667;AA=T;DB GT:GQ:DP:HQ 1|2:21:6:23,27 2|1:2:0:18,
20 1230237 . T . 47 PASS NS=3;DP=13;AA=T GT:GQ:DP:HQ 0|0:54:7:56,60 0|0:48:4:51,51 0/0:61:2
20 1234567 microsat1 GTC G,GTCT 50 PASS NS=3;DP=9;AA=G GT:GQ:DP 0/1:35:4 0/2:17:2 1/1:40:3
```

The first thing you notice is that it describes only variant positions **NOT EVERY** position like Pileup.

The lines starting with `##` are the meta-data description lines. They define the labels for SNP data and contain key-value pairs separated by an '=' sign (e.g. number=1, Type=Integer, Description="Some description").

The header line starts with a single `#`, and has the column headings, these being `ALT` and `QUAL`.

1. `CHROM` = chromosome
2. `POS` = position
3. `ID` = an ID (if given)
4. `REF` = reference sequence nucleotide
5. `ALT` = SNP nucleotide
6. `FILTER` = a filter defined in the metadata
7. `INFO` = Summary information about this SNP
8. `FORMAT` = Format of the sample column(s)

---

[1] https://samtools.github.io/hts-specs/VCFv4.2.pdf

9. Sample information for one sample formatted according to `FORMAT`
10. More sample info (if needed), also according to `FORMAT`

`INFO` and `FORMAT` are where the VCF format really makes use of its meta-data. In the INFO column there will be a combination of key=value pairs, separated by semi-colons. An entry will read something like:

```
ADP=27;WT=0;HET=2;HOM=0;NC=0
```

Each key (e.g. `ADP`) will have its meaning explained in the metadata `INFO` fields.

The `FORMAT` column reads slightly different from the `INFO` field.

```
GT:GQ:SDP:DP:RD:AD:FREQ:PVAL:RBQ:ABQ:RDF:RDR:ADF:ADR.
```

The meaning of these keys can be found in the meta-data in the `FORMAT` lines. Some common ones that are useful are:

- `DP` = coverage depth
- `GT` = genotype, e.g `0/0`, `1/1` and `0/1`

Genotype values have the following meanings:

- `0/0` - Homozygous to the reference (`REF`)
- `1/1` - Homozygous to the alternate non-reference allele (`ALT`)
- `0/1` - Heterozygous (`0/2` represents a heterozygote with two alternate alleles)

Deciding on the right parameters for SNP calling is very case specific. Often SNP calling pipelines will need to be optimised to ensure that the level of false positive calls is acceptable. Here are some parameters in SNP callers to look out for

- Minimum coverage - the number of read bases that contribute to the SNP call, more is better
- Minimum variant allele - the fraction of bases in the pileup that are needed for a SNP call (1/100 = bad, 50/100 = believable)
- P-value threshold - or some other probability based measure of SNP accuracy.
- Strand filter - discard SNPs where most reads come from one strand - helpful with certain sequencing errors
- Base quality - discard bases in the pileup that don't have good enough individual Phred scores

Whichever tool you use, make sure that you get a good idea of what it's options for making high-quality SNPs are.

4.3 Exercises

4.3.1 MPileup

You are provided with a new BAM file for *Arabidopsis* chromosome 4 in the shared data library `SNP Calling`, use it to run `HTS SAMtools .. SAMtools mpileup`. Remember you'll need to load a reference genome, the `TAIR_10_chr4.fasta` file should be imported to your history for this purpose.

1. How do you make `SAMtools mpileup` output VCF or Mpileup?
2. What are differences in information between the two?

3. Generate an MPileup file and select appropriate `advanced options` to make sure you have a good enough mapping quality (~20) and base quality (~30) for reliable SNP calls.

4. What is the point of adding a maximum read depth?
5. How does this run compare in execution time with earlier ones in this course? Why is there a difference?

These are the largest datasets we use in this workshop The reference is a whole 20 Mb *Arabidopsis* chromosome, with full 30 deep covereage. The laptop VM takes a while to chug through it. If you are getting bored, you can restrict the amount of the reference that SAMtools will churn through. Try looking in `Advanced Options` for `Select regions to call`. You can paste or type in a region in the format `Chr4:1-100` - replacing the 1 and 100 with suitable start and stop coordinates. Try doing just 2000000.

4.3.2 VarScan

From the mpileup file you created in the challenge above, use `VarScan Mpileup` in `Finding Variants` to filter the positions to find the SNPs and make the criteria a bit more stringent.

1. Inspect the pileup (or run some SAMtools stats) to determine suitable values for the depth and quality parameters.
2. Our purpose is to clearly separate Homozygous and Heterozygous SNPs, which filters do you think we should use? What frequency values should we set to get many, good homozygous calls (hint: 100% frequency rules out some SNPs where a single miscalled read or base messes things up)
3. How long does this take? What factors do you think affect the run time?

Chapter 5

Visualising SNPs To Find Candidates

5.1 Learning Objectives

- Understand CandiSNP output
- Plotting frequencies of SNPs of different types across the chromosome

5.2 Annotation

Once we've called SNPs, then the next stage is to annotate them. We are looking for heterozygous and homozygous information, which we get from the SNP callers, but a phenotype causing SNP is much more likely to cause a change in the protein that is encoded. Thus we need to know what the effect of the SNP on the protein is.

These qualities are what we will look for in our candidate SNP approach - the most likely mutation causing SNPs will be:

- Homozygous
- Non-synonymous
- In a region enriched with homozygous SNPs

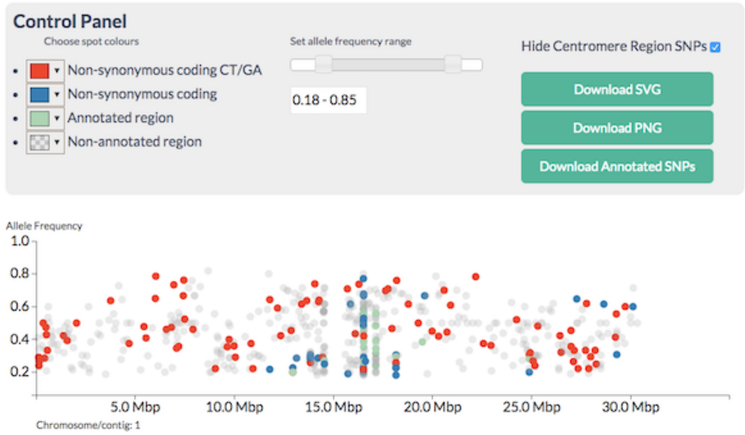
5.2.1 SNPEff

We can annotate SNPs with a program called `SNPEff`. SNPEff (Cingolani et al., 2012) is a fairly straightforward system. It needs a database of SNPs and a database of gene and gene/transcript positions and can give you whether the SNP is in a gene and whether the SNP causes a silent synonymous[1] change (IE the codon the SNP changes is for the same amino acid before and after the SNP sequence change) or whether it causes a more significant non-synonymous[2] change that actually changes the amino acid at that codon so the protein changes.

---

[1] https://en.wikipedia.org/wiki/Synonymous_substitution

[2] https://en.wikipedia.org/wiki/Nonsynonymous_substitution


Control Panel
Choose spot colours
🟥 Non-synonymous coding CT/GA
🟦 Non-synonymous coding
🟩 Annotated region
▨ Non-annotated region
Download SVG
Download PNG
Download Annotated SNPs
Allele Frequency
1.0
0.8
0.6
0.4
0.2
5.0 Mbp
10.0 Mbp
15.0 Mbp
20.0 Mbp
25.0 Mbp
30.0 Mbp
Chromosome/contig: 1

Set allele frequency range: 0.18 - 0.85

Hide Centromere Region SNPs ☑

Figure 5.1: Screenshot of the CandiSNP tool. Spots represent SNPs (height on the y axis shows major allele frequency).

Once we have a list of SNPs that we are happy with and have annotated them with SNPEff, there are a couple of approaches we can take to start finding candidates that may be our causative mutation.

The approach we take will depend on the genetic background. As we discussed at the start, we are generally looking for a region of high homozygous SNPs, but the frequency of other SNPs will depend on the cross. A wide cross from a fairly distant relative (like a different strain or ecotype) as is commonly used in genetic mapping strategies will allow us to make use of the heterozygous SNPs as a control.

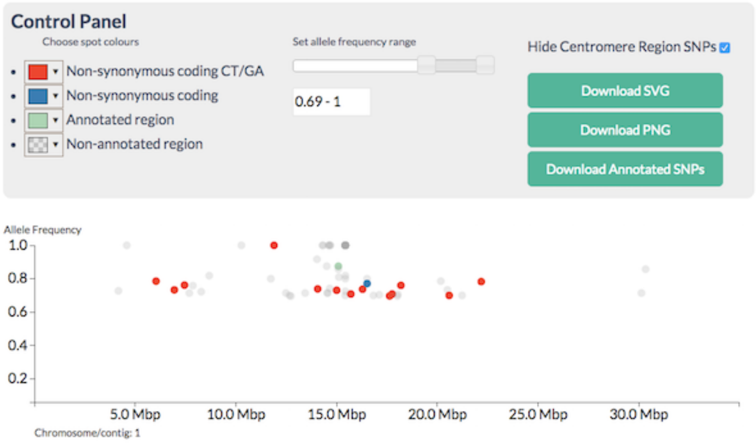
5.3 CandiSNP

Fast interactive visualisations are a great help in finding the recombinant region and narrowing the candidates. One tool that allows us to do this is `CandiSNP`. CandiSNP (Etherington et al., 2014) is a JavaScript visualisation package that allows interactive filtering and highlighting of SNPs across whole chromosomes (DISCLAIMER: My group wrote this!). The tool isn't available inside Galaxy, but is on the web at http://candisnp.tsl.ac.uk. A significant advantage of CandiSNP is that is has SNPEff built into it. So you can use the unannotated SNP file straight in CandiSNP.

CandiSNP allows you to look at the SNPs like in Figure 5.1 and apply filters to narrow down the region and candidates so you see something like Figure 5.2. CandiSNP takes a VCF file as input.

5.4 Density plots

Statistical methods are useful when the number of SNPs generated is so large that you can't visualise them all at the same time. Density plots like Figure 5.3 (which is of the same data as the CandiSNP plot in 5.1 help us to see the rough patterns in a similar way. The homozygous and heterozygous show an increase in the SNP-rich centromeric region which biases the data and an overall decrease at the far right of the chromosome, but the enriched region is visible in the high ratio at about 17Mbp as in the CandiSNP output. These kinds of plots can be generated with Galaxy's plotting tools.



Set allele frequency range 0.69 - 1

Hide Centromere Region SNPs ☑

Control Panel
Choose spot colours
🟥 Non-synonymous coding CT/GA
🟦 Non-synonymous coding
🟩 Annotated region
▨ Non-annotated region
Download SVG
Download PNG
Download Annotated SNPs
Allele Frequency
1.0
0.8
0.6
0.4
0.2
5.0 Mbp
10.0 Mbp
15.0 Mbp
20.0 Mbp
25.0 Mbp
30.0 Mbp
Chromosome/contig: 1

Figure 5.2: CandiSNP after filtering. The region of the high red spot density is the recombinant region

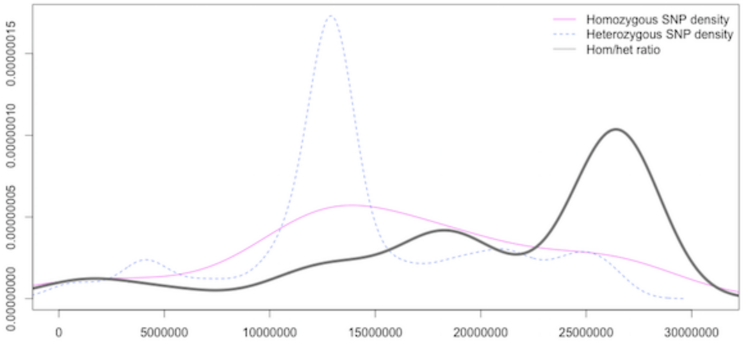


Figure 5.3: Density plot of homozygous, heterozygous SNP density and the ratio of hom/het SNPS in sliding windows

5.5 Centromeres

Centromeres are a real problem with these sorts of analysis. They are so SNP rich that they swamp analysis and visualisations. It helps to just screen them out from the analysis. CandiSNP will let you turn off centromeres associated SNPs, Galaxy tools can also help you filter them out.

5.6 SNP Deletion - Fewer are better

Perhaps it is counter-intuitive but getting fewer SNPs is often better in these approaches. A common source of confounding SNPs is from the parental line itself. All individuals of any species have differences in the genomes from the references we use to call SNPs, and some (perhaps many) of these will be shared between the parent used to generate the mutants and the mutants. By sequencing the parent line and calling SNPs between it and the reference genome, you get a list of parental SNPs that you can often delete straight out of the mutant as being non-causative.

5.7 Exercises

5.7.1 Analyse SNP data with CandiSNP

You have some whole genome *Arabidopsis* SNP data annotated with SNPEff in the shared data library `Annotation`, the VCF file `filtered_snps.vcf`. Use this in the web version of CandiSNP tool at [] This data set is a real one and we know exactly where the mutation is because we've sequenced it, so there is a *right* answer. Use the sliders and filter tools to find a region enriched in homozygous candidate SNPs.

1. Can you come up with candidate regions / genes for the causative mutation?
2. Which is more useful, filtering or colouring?
3. How much extra information does knowing the genes the SNPs fall in give? In this specific training case and in a real case where you might know something about the biology already.

5.7.2 Generate density plots of different SNP classes

Standard Galaxy tools can generate histograms of data. However the data needs to be in tabular format, not VCF. Here's a little recipe for going from VCF to a table that is useful.

1. To make a tabular file, use the `Text Manipulation .. Cut Columns Tool`. Cut out columns `c1,c2,c3,c4,c8,c9`.
2. To strip text from the AF field and just leave the numbers, use the `Text Manipulation .. Trim Leading` on `column 5`, trim to position `4` and set to `ignore #`.
3. To get the homozygous SNPs, use the `Filter and Sort .. Filter data on any column` tool. Filter on `c5>=0.75` (or whatever seems sensible to you).
4. To get the heterozygous SNPs, do step `3` again but filter on `c5<0.75`.

This will leave you with two files on which to carry out the remaining steps.

5. To split the files into single chromosome files use `Text Manipulation .. Split file according to value of a column` use `column 1` (the chromosome column)
6. For each resulting file you can use `Plotting .. Histogram` to make the histograms. It is useful to add the density plot and to use around 150 breaks.

With a bit more work you can combine the different plots into one large one for easier comparison, and you can use sliding window tools to calculate the ratio of Hom/Het SNPs across the chromosomes.

5.7.3 Comparing density plots

1. Use the numerous density plots you made to compare the likely positions of the causative SNPs
2. Can you narrow down the area sufficiently to examine the text list in more detail?
3. What further filtering could you do to make the plots more effective?

Bibliography

Andrews, S. (2017). *FastQC: A quality control tool for high throughput sequence data.*

Bolger, A. M., Lohse, M., and Usadel, B. (2014). Trimmomatic: a flexible trimmer for Illumina sequence data. *Bioinformatics (Oxford, England)*, 30(15):2114–2120.

Cingolani, P., Platts, A., Wang, L. L., Coon, M., Nguyen, T., Wang, L., Land, S. J., Lu, X., and Ruden, D. M. (2012). A program for annotating and predicting the effects of single nucleotide polymorphisms, SnpEff: SNPs in the genome of Drosophila melanogaster strain w1118; iso-2; iso-3. *Fly*, 6(2):80–92.

De Summa, S., Malerba, G., Pinto, R., Mori, A., Mijatovic, V., and Tommasi, S. (2017). GATK hard filtering: tunable parameters to improve variant calling for next generation sequencing targeted gene panel data. *BMC bioinformatics*, 18(Suppl 5):119.

Etherington, G. J., Monaghan, J., Zipfel, C., and MacLean, D. (2014). Mapping mutations in plant genomes with the user-friendly web application CandiSNP. *Plant methods*, 10(1):41.

Koboldt, D. C., Chen, K., Wylie, T., Larson, D. E., McLellan, M. D., Mardis, E. R., Weinstock, G. M., Wilson, R. K., and Ding, L. (2009). VarScan: variant detection in massively parallel sequencing of individual and pooled samples. *Bioinformatics (Oxford, England)*, 25(17):2283–2285.

Li, H. and Durbin, R. (2010). Fast and accurate long-read alignment with Burrows-Wheeler transform. *Bioinformatics (Oxford, England)*, 26(5):589–595.

Li, H., Handsaker, B., Wysoker, A., Fennell, T., Ruan, J., Homer, N., Marth, G., Abecasis, G., Durbin, R., and Subgroup, . G. P. D. P. (2009). The Sequence Alignment/Map format and SAMtools. *Bioinformatics (Oxford, England)*, 25(16):2078–2079.

Page, D. R. and Grossniklaus, U. (2002). The art and design of genetic screens: Arabidopsis thaliana. *Nature reviews Genetics*, 3(2):124–136.