Master Project

Lab book

Cyril Matthey-Doret

December 14, 2017

Chapter 1

Introduction

The goal of this project is to identify the locus or loci responsible for sex determination in the parasitoid wasp *Lysiphlebus fabarum*. The species is known to have CSD, which means there are one or more loci that need to be heterozygous in order to triggger female development. I will use the offspring of asexual (thelytokous) females to identify these loci. The offspring consists of haploid males, diploid males and diploid females. Using DNA sequencing data, I will exclude haploid males as they do not provide information on homozygosity/heterozygosity and compare only diploid males versus females to identify the CSD locus/loci. *Lysiphlebus fabarum* wasp specimens (generation F4) issued from thelytokous mothers (generation F3) are used. These individuals come from crossing experiments in a strongly inbred line. Here, we use restriction site associated DNA sequencing(RAD-seq) with a custom pipeline to locate the locus/loci. Samples were single-end sequenced using a ddRAD protocole and digested with ecoRI and mseI. There are 2 separate libraries with two different illumina adaptors (detailed in next section). Statistics for the raw RAD-seq data of both libraries of F4 individuals are shown in table 1.1. I also reused sequencing data for the mothers (F3) from Casper, however I don't have the raw reads statistics as they were already processed.

Data summary:

library	lib7	lib7b
raw reads	163,506,603	133,574,055
containing adaptors	23.25%	45.84%
fragment size	302bp	302bp
mean quality score	34.88	35.05
>= Q30 bases	92.62%	92.13%

Table 1.1: Summary statistics of raw reads from the 2 libraries of F4 individuals RAD-seq data.

This lab book will document my progression for mapping the CSD locus, which can be broken down into 4 main steps: preprocessing of sequencing data, STACKs analysis for RAD-seq data, association mapping and orthology search. These steps are likely to change and throughout the project and it is likely that more will be added. The general process is described visually in figure 1.1.

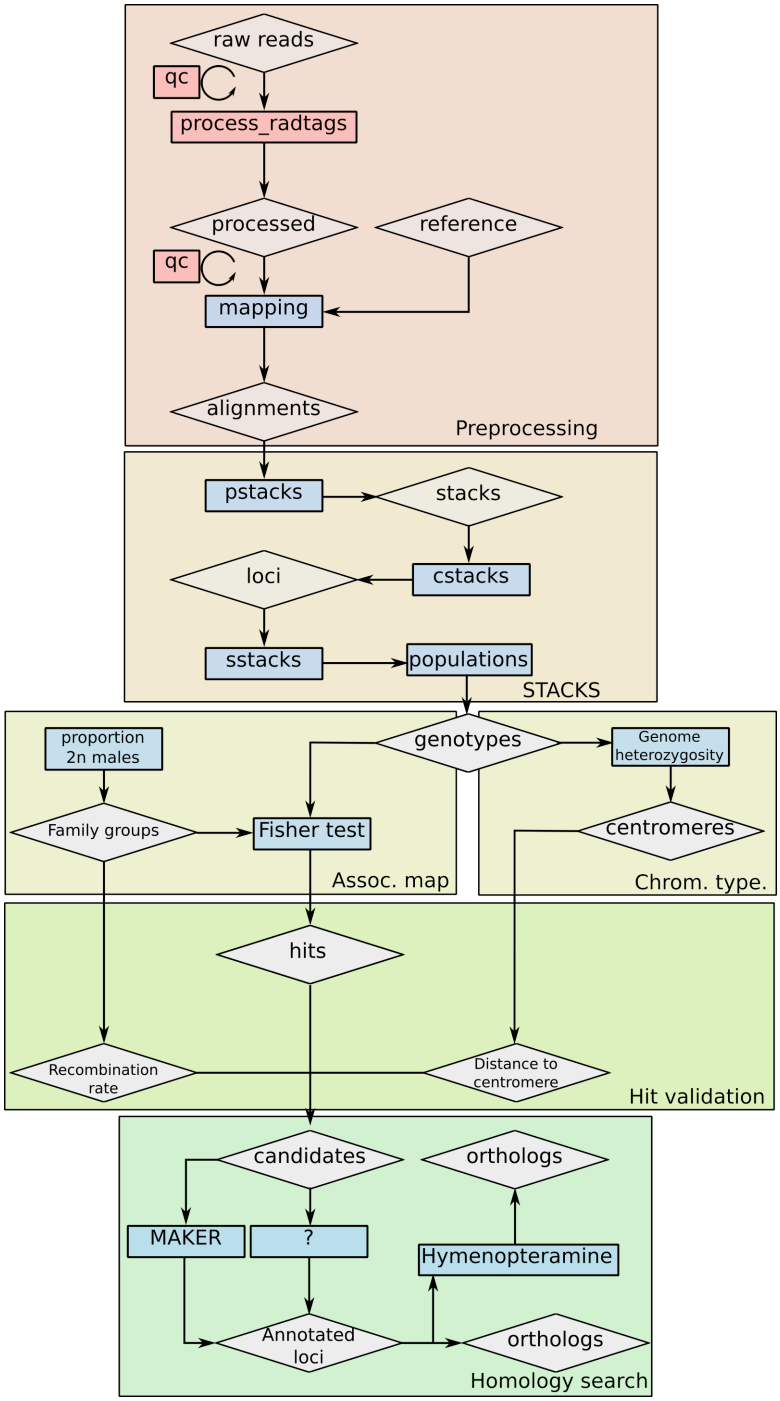

Figure 1.1: **Pipeline described in the current lab book**. Diamonds represent data, rectangles represent operations/programs. Operations/programs in blue are included in the makefile, thos in red are not.

Chapter 2

Processing reads

RAD-seq data was split into 2 separate libraries: 7 and 7b. Together, the libraries contain 173 F4 individuals from 11 different F3 mothers. There were 96 samples in library 7 (one of which was contaminated) and 77 in library 7b. In total we have 172 valid samples across 11 families. Additionally, I inherited from 28 individuals from another library (lib6) which are offspring from families A and B.

2.1 Quality control

fastqc was used for quality control, separately on each file, and on all files together in the library.

2.2 Demultiplexing

The process-radtags program from stacks was used for demultiplexing and removal of Illumina adaptors. The operation was performed separately for libraries 7 and 7b:

```
$ process_radtags -p raw/ -o processed/ -b \
/barcodes -e ecoRI --filter_illumina -E phred33 \
 -r -c -q --adapter_1 adapter --adapter_mm 2
```

lib7	Truseq adapter, index 6 [1]
lib7b	TruSeq adapter, index 12 [2]

2.3 Trimming adaptors

This step is performed by process radtags at the same time as demultiplexing. I tried different values for adapter mismatches, between 0 and 3 (i.e. reads containing sequences distant from the adapter by n mismatches are removed). This did not cause any major difference and therefore I will perform all downnstream analyses with an allowance of 2 mismatches in the adapter.

Note: Demultiplexing did not yield any read for CF4F10.

[1] GATCGGAAGAGCACACGTCTGAACTCCAGTCACGCCAATATCTCGTATGCCGTCTTCTGCTTG
[2] GATCGGAAGAGCACACGTCTGAACTCCAGTCACCTTGTAATCTCGTATGCCGTCTTCTGCTTG

Chapter 3

STACKS pipeline

The pipeline described in this chapter will map the processed reads to the reference genome and build a catalogue of loci. It will eventually implement additional features such as calculating population statistics. At each step of the pipeline, I will try different combinations of parameters and choose the one yielding the best results.

3.1 Mapping

Since the reference genome of *Lysiphlebus fabarum* was recently released, I will first map the sequencing reads to the reference, using BWA, but I may also want to use bowtie to compare the results, eventually. At the moment, a draft reference genome is available, table 3.1 displays some summary statistics of it.

3.1.1 BWA

BWA provides 3 different algorithms: MEM, backtrack (aln) and SW. MEM is normally the better for Illumina reads longer than 70bp, therefore I will be using this one here. Backtrack is preferred for short reads and SW with frequent gaps. There are also 2 algorithms for building the index: 'is' and 'bwtsw'. 'is' is used with reference ¡2GB, 'bwtsw' with larger references.

General commands:

```
$ bwa index -p <out_index_name> -a <algorithm>
$ bwa mem <index> <sample.fq> > <out.sai>
> $sample-$prefix.sam
```

When using bwa-aln, there the command running the alignment (after indexing) is:

```
$ bwa aln -n <mismatches> $index <sample.fq> > <sample.sai>
$ bwa samse -n <max_dupl> $index $sample.sai $data_dir/$sample.fq.gz
```

The first command (bwa index) constructs an index from the reference genome, whereas the second one (bwa mem/aln) actually runs the aligner. The third command (bwa samse) allows to transform the .sai into .sam files.

Statistics	Values
Assembly length (Mbp)	140.7
Largest contig (Mbp)	2.2
Mean contig size (kbp)	82.9
Median contig size (kbp)	30.4
N50 (kbp)	216.1
Number of contigs	1698

Table 3.1: Assembly statistics for the current reference genome of *Lysiphlebus fabarum*

Here is the list of different mapping parameters I may try with bwa-mem:

- -k : minimum seed length (will miss matches shorter than value)[19]
- -w : band width (gaps longer than value will not be found)[100]
- -d : maximum distance between query and reference positions before stopping seed extension. [100]
- -r : triggers reseeding for a MEM longer than min_seed_len*float. Larger values yield fewer seeds − > faster alignment but lower accuracy [1.5].

In the case we use the backtrack (aln) algorithm instead of MEM, the only parameter worth tuning is -n, the number of mismatches allowed.

Note: I did not include parameters that are not relevant to sensitivity (e.g. threads) or parameters that involve scoring (changing these naively would probably have a negative impact). Full list of parameters is on the official bwa website

Note2: About multiple hits and BWA-mem:
(https://github.com/lh3/bwa)
2. Why does a read appear multiple times in the output SAM?

BWA-SW and BWA-MEM perform local alignments. If there is a translocation, a gene fusion or a long deletion, a read bridging the break point may have two hits, occupying two lines in the SAM output. With the default setting of BWA-MEM, one and only one line is primary and is soft clipped; other lines are tagged with 0x800 SAM flag (supplementary alignment) and are hard clipped.

3.1.2 Bowtie2

General commands:

```
$ bowtie2-build reference_genome.fa L_fabarum
$ bowtie2 -x <ref_index> -U <unpaired_reads_files> -S <out_SAM_file>
```

The first command (bowtie2-build) constructs a set of index (extension: .bt2) from the reference genome, whereas the second one actually runs the aligner.

Here is the list of different mapping parameters I may try:

- –trim5 ¡n¿ : trim n bases from the 5' end (left) of each read before alignment
- –trim3 ¡n¿ : trim n bases from the 3' end (right) of each read before alignment
- -D : maximum number of seed extension that can fail in a row before stopping (increasing makes bt2 slower)
- -R : maximum number of re-seeding when attempting to align read with repetitive seeds (increasing makes bt2 slower)
- -N : number of mismatches permitted per seed (increasing reduces false negative, but makes bt2 slower)
- -L : length of seeds (decreasing makes bt2 slower but more sensitive)
- -i : interval between seeds (increasing makes bt2 slower but more sensitive)

Recommandations from the bowtie2 website to make alignment more sensitive: a) make seeds closer (reduce i) b) make seeds shorted (reduce L) c) allow more mismatches per seed

End-to-end versus local alignment (–local): end to end takes all bases in the reads into account, while local allows to trim reads to exclude the ends from the alignment. Seeds are substring of the reads which bowtie2 tries to align to narrow down the valid regions for aligning a read. There is a list of preset values for these parameters on the official bowtie2 website. Preset parameters differ between local and end-to-end modes. There are other parameters, such as score weights for gaps mismatches and allowance for 'N' ambiguous characters but changing these naively could probably have negative effect on the alignments. Procedure: try out different preset values and select the one yielding the best results. Then, eventually tweak the parameters slightly from the preset values. These simulations can be run on a subset of individuals to speed up the process.

note to self: at the end of a run, bowtie2 prints a summary to stderr such as:

20000 reads; of these:
20000 (100.00%) were unpaired; of these:
1247 (6.24%) aligned 0 times
18739 (93.69%) aligned exactly 1 time
14 (0.07%) aligned >1 times
93.77% overall alignment rate

If I use Bowtie2, I will redirect the stderr and parse it into a csv file and generate plot to visually estimate the best parameter values.

3.1.3 Mapping results

I used the aln algorith of bwa, since the mem algorithm did not report multiple alignments and has very few documentation available. I tried different mismatches values with aln (Figure 3.1), ranging from 0 to 8, and I chose to use 4 since the increase in single was quite low above this value. When the mismatch parameter is set to 4, running bwa-aln on a subset of 12 samples yielded 57% of single hit reads, 26% of multiple hits and 17% of unmapped reads.

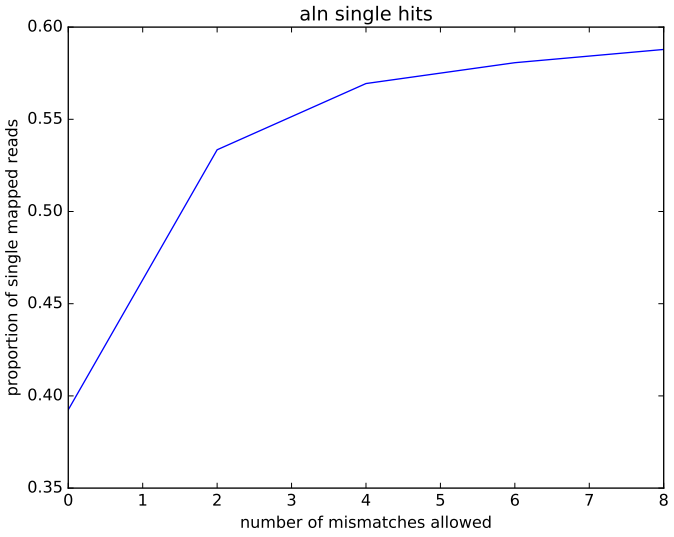

Figure 3.1: Results of the BWA mapping with different parameter values for the number of mismatches allowed during mapping. The plot shows the proportion of reads that are mapped uniquely to the reference genome.

3.2 pstacks

pstacks is a component of the STACKS suite that takes stacks of reads aligned to a reference genome as an input (typically in the SAM format) and idenify SNPs

general command:

```
$ pstacks -f <input_path> -i <sample_ID_int> -o <out_dir> \
 -m <min_depth> -p <num_threads> -t <file_type>
```

Parameters I may want to change are:

- -m : minimum depth of coverage required to call a stack [3]
- –max_clippped : alignments with more than X soft-clipped bases are discarded [15%]
- –min_mapq : minimum required quality [10]

min˙cov	nloci	mean˙cov	sd˙cov
1	5020.1	31.8	58.6
2	3659.1	42.5	65.0
3	2747.2	49.9	68.2
4	1505.3	55.6	70.1
5	781.9	60.1	71.3
6	597.9	63.9	72.3

Table 3.2: Summary statistics of stacks obtained with different parameter values for minimum coverage in pstacks.

Note: there are 3 models; SNP, bounded SNP and fixed. SNP is the default model, bounded SNPs allows to give prior expectations about the error rate, which can allow better estimations of heterozygosity and the fixed model identifies all fixed sites and masks all others.

3.2.1 pstacks results

Pstacks was run on aligned reads (BWA, 4 mismatches allowed). I tried different values for the minimum coverage required to call a stack (-m parameter), ranging from 1 to 6 (Figure 3.2). Below is the value for minimum coverage, along with the mean number of loci and alleles that was produced per sample. This table was produced including all non-empty samples (198 individuals) and the variables are averaged (arithmetic mean) over all those samples.

I will use 4 reads minimum coverage as this is already high and using lower values does not improve the output in anyway.

Note: Locus are regions formed by one or more stacks. Alleles are different stacks at the same locus.

3.3 cstacks

cstacks is a component of the STACKS suite that builds a catalog of loci with different alleles from a set of processed samples.

general command:

```
$ cstacks -s <sample_prefix> -o <out_dir> -b <catalogue_ID>\
-p <num_threads> -n <num_mm> -M <pop_map>
```

Parameters I may want to change are:

- -n : Number of mismatches allowed between sample loci when building catalogue [1]
- -g : base catalog on alignment position instead of sequence identity

Note: There are also advanced options such as gapped assembly parameters and loci matching multiple catalogue entries, but these are probably not relevant here.

3.3.1 cstacks results

I changed the number of mismatches allowed between samples between 1 and 4. Table 3.3 shows the results with a subset of 11 samples ($[A-L\&\&[\wedge B]]01$) because B01 was empty (few, low quality reads).

I tried running cstacks on all (non-empty) samples, and also modifying the script so that it will first compute the mean number of tags (reads) across all samples, and exclude those with less than 10% of this value when building the catalogue. From the 202 original samples, 4 were empty and excluding low quality ones removed 13 additional ones.

mismatch	mean˙loci	mean˙alleles
1	2696	7799
2	3129	8464
3	3198	8704
4	3263	8914

Table 3.3: Summary statistics of loci obtained with different parameter values in cstacks.

Update: 21.07.2017
When including mothers, I reach a total of 212 individuals. I use the same filtering strategy to exclude samples with low amounts of reads (Figure 3.2). From the 212 original samples, I excluded 17 low quality samples, among which 4 were totally empty.

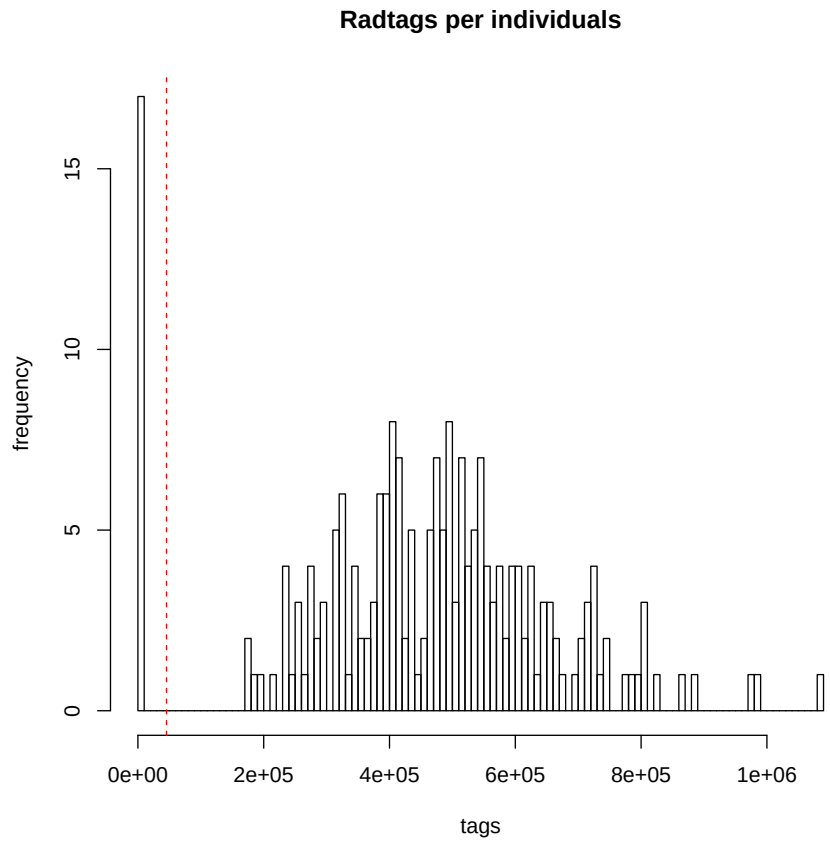

Figure 3.2: distribution of the number of radtags (reads) per sample. The red vertical dashed line indicates the cutoff value set to 10% of the mean. Samples below this value are removed from the analysis.

Summary results with n=1:

- All non-empty samples (198): 25618 alleles in catalogue, over 7062 loci.
- Excluding samples with <10% of mean tags (185): 25585 alleles in catalogue, over 7046 loci

note: following recommandations from Paris et al. 2017, Lost in parameter space: a roadmap for STACKS. *Methods in Ecology and Evolution*, I set the value of n to 3 (M-1=3)

3.4 sstacks

sstacks is used to generate one file per individual, in each file, the matching loci point to the cstacks catalogue. There is no crucial parameter to change in this program.

3.5 populations

The "populations" component of STACKS is used to compute population genetics statistics on a set of individuals. I use it here to compute FST statistics (fixation index) along the genome. It offers several features to compute different statistics, including a bootstrapping feature and a "kernel smoothing" flag, allowing to take neighbouring region into account with a decreasing weight as a function of their distance from the focus nucleotide. I will use both of these features to compute FST.

The main features to change in FST calculation are :

- -r: minimum percentage of individuals in a population required to process a locus for that population.
- -p: minimum number of populations a locus must be present in to be procesed.
- -m: minimum stack depth required for individuals at a locus.

Example populations call for FST calculation:

```
$ populations -P <stacks_files> -M <popmap> -b 1
 -k -r 0.75 -f p_value
```

3.5.1 populations results

I ran populations with the following parameters:

- -r: 0.75 - 0.85
- -p: 2
- -m: 5

Therefore, only loci with at least 5 reads of coverage were included, each loci also needs to be in at least 75% to 85% of all individuals and in both populations (males and females).

This table summarizes the first statistics I extracted from the VCF files (using vcftools) with the different values for r:

r parameter	obs.hom.	exp.hom.	n.sites	inbreed.coef.	mean depth	obs./exp. hom.
75	957.89	895.09	1171.82	0.23	121.09	1.07

plotting the inbreeding coefficient per individual with r=75 yields:

When increasing the minimum depth above 15, populations crashed. Disabling bootsrapping and kernel smoothing fixed the issue. Guess: May be caused by a contig smaller than the sliding window size during kernel smoothing. This could be the case if the sliding window has a min number of loci, in which case increasing minimum depth would cause the window to enlarge.

I tested for correlations between Mean depth of loci and homozygosity in all individuals and per family, no significant correlation was found. This means there should not be significant allele excludion caused by too stringent min depth (i.e. stacks removed because of low coverage, resulting in homozygous loci).

3.6 STACKS parameters summary

process radtags	Mis:2
bwa	Mis:4
Pstacks	MinDep:3
Cstacks	LocMis:1
Sstacks	-
populations	IndProp:0.75, MinDep:5(25)*, MaxHet:0.9

* Stringent MinDepth value used to exclude haploid males. The smaller value is used for downstream analyses to keep more loci.

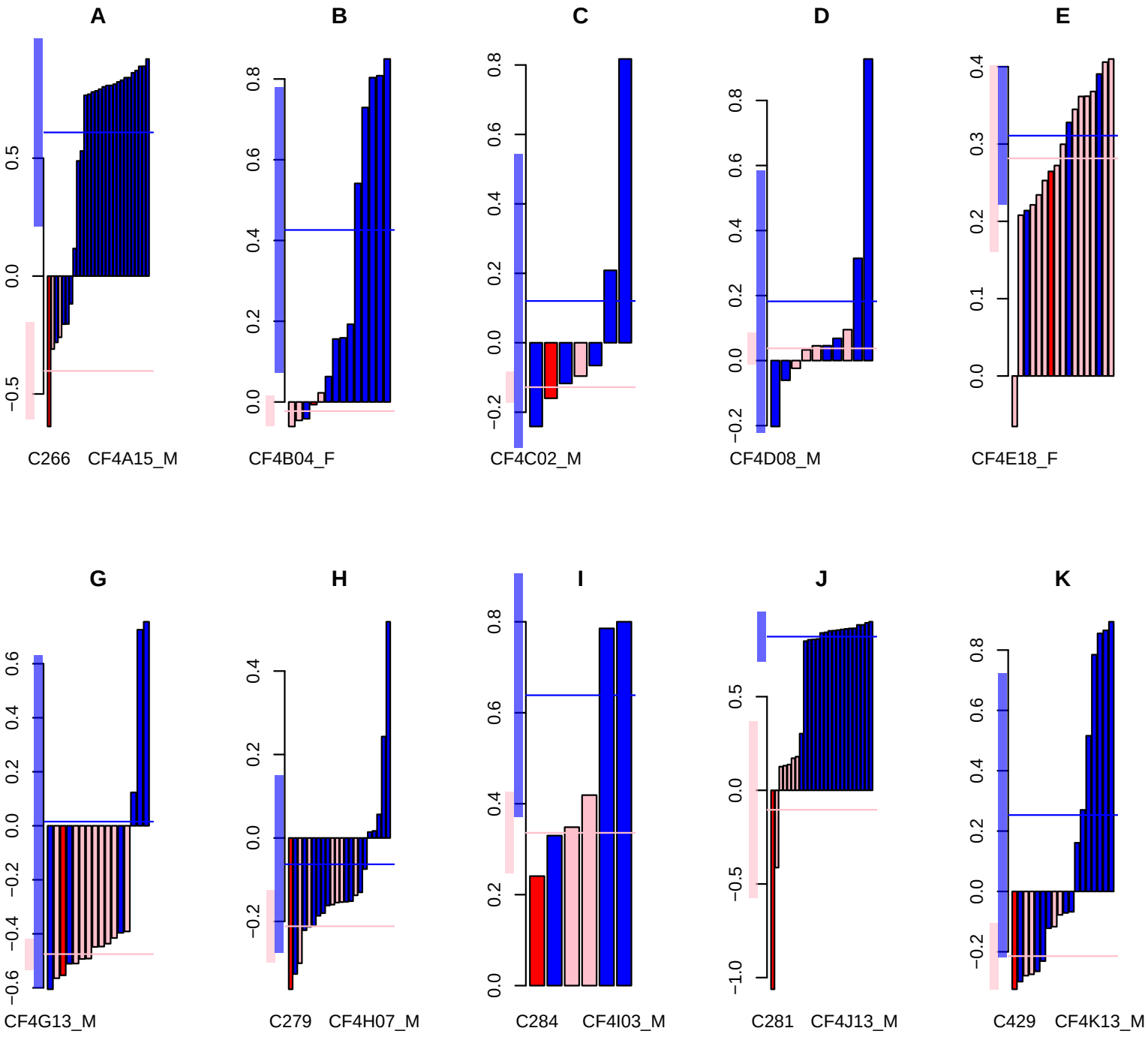

Figure 3.3: Inbreeding coefficient (F). Each plot is a family, each bar is an individual. Blue bars represent males and pink ones represent females and red ones are mothers. color bars on the y axis span the mean +- standard deviation of males and females, respectively. In theory, mothers should have the lowest inbreeding coefficient of their family (highest heterozygosity)

Chapter 4

Excluding haploids

I used the F statistics (coefficient of inbreeding) computed by vcftools to measure homozygosity levels in each individual. The F statistics used were generated with more stringent parameters (min depth: 25) to call ploidy more confidently. It is calculated as $F = \frac{O-E}{N-E}$ where O is the observed number of homozygous loci, E is the number expected by chance and N is the total number of loci.

First, I calculated the Mean μ and standard deviation σ of the inbreeding coefficient F among daughters in each family. I then classified males with:

$$F > \mu + 2\sigma$$

as haploid. This is stringent threshold should exclude all haploid males, but it can also potentially exclude diploid males from the analysis.

Update: 08.06.2017

I tried 12 different thresholds for excluding haploids. The thresholds were computed according to these rules:

- 1: All thresholds follow the formula $\mu + \tau(N\sigma)$ with $1 \leq N \leq 4$
- 2: $1 \leq \mu \leq 4$
- 3: τ is a function for transforming the inbreeding coefficient values. Either $\tau(F) = F^2$ or $\tau(F) = \sqrt{F}$

As shown above, in this report I will use the threshold named "m2" which corresponds to $F > \mu + 2\sigma$. Figure 4.1 shows a plot showing the separation of haploid and diploids using this threshold.

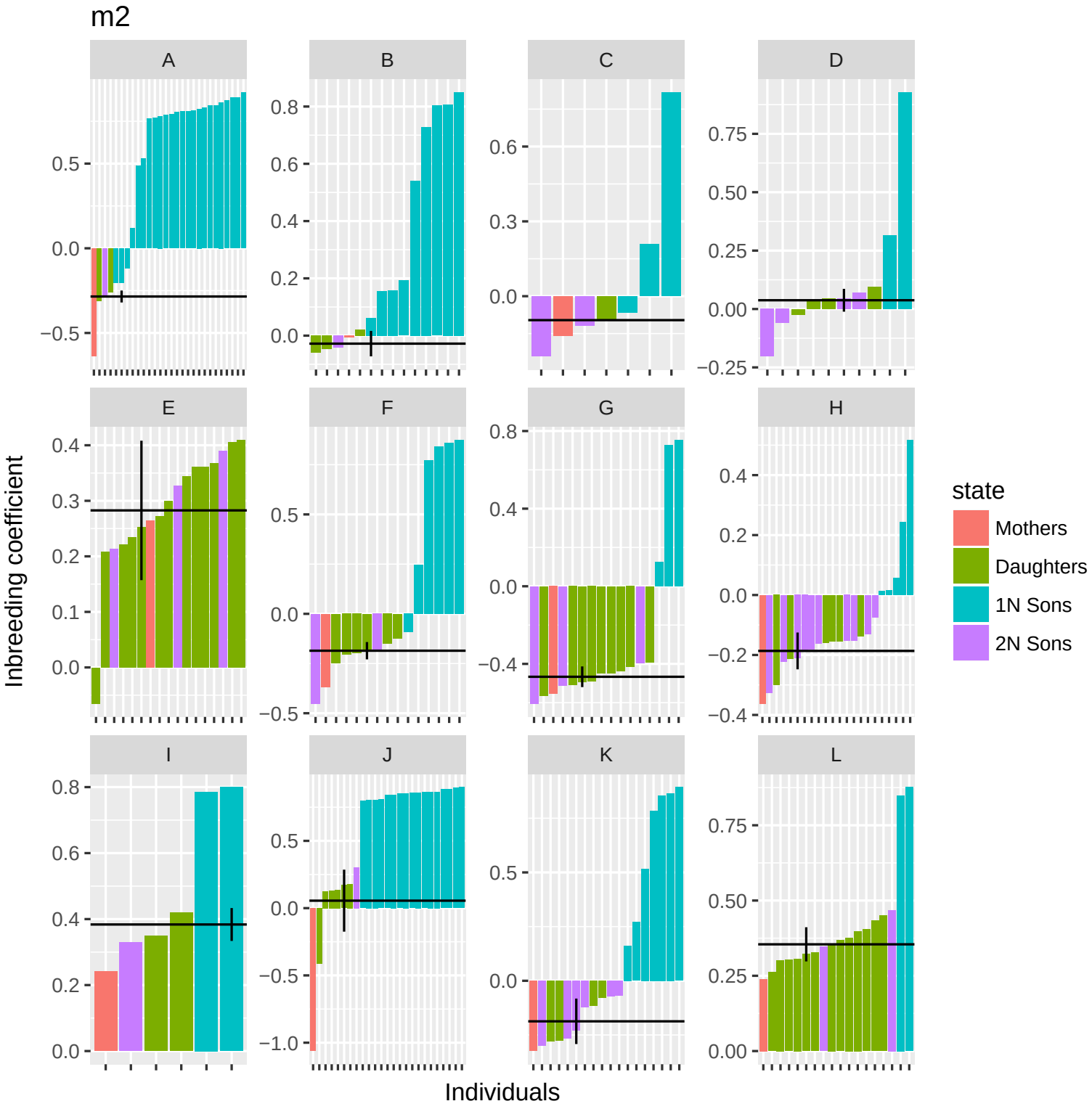

Figure 4.1: Inbreeding coefficient (F). Different colors represent the different types of individuals, as shown in the legend. Each plot is a family, each bar is an individual. Horizontal black bars show the mean inbreeding coefficient of daughters within the family +/− the standard deviation (vertical black line). In theory, mothers should have the lowest inbreeding coefficient of their family (highest heterozygosity)

After separating haploids from diploids, I performed exploratory analyses prior to association mapping. The goal of these is to assess the homozygosity rate of SNPs in diploid males and females and how many seem to fit the CSD pattern. As shown in Figure 4.2 and 4.3, there is no single SNPs thatis heterozygous in all females and homozygous in all diploid males. That can imply that the species has ml-CSD and there is no single locus that always fit the CSD pattern, or the restriction enzyme used in the RAD-seq protocol (ecoR1) did not cut in the CSD locus directly. It is very likely a combination of both explanations. It is also worth mentioning that, besides the number of SNPs, it makes little difference whether I use permissive (Figure 4.2) or stringent (Figure 4.3) parameters. The only notable change is that including more SNPs results in higher homozygosity for both sexes.

In this analysis, I excluded all consensus loci (where all individuals had the same allele) and did not consider SNPs with more than 2 genotypes in a single individuals in the heterozygosity calculation (counted as missing values), as they are likely due to contaminations.

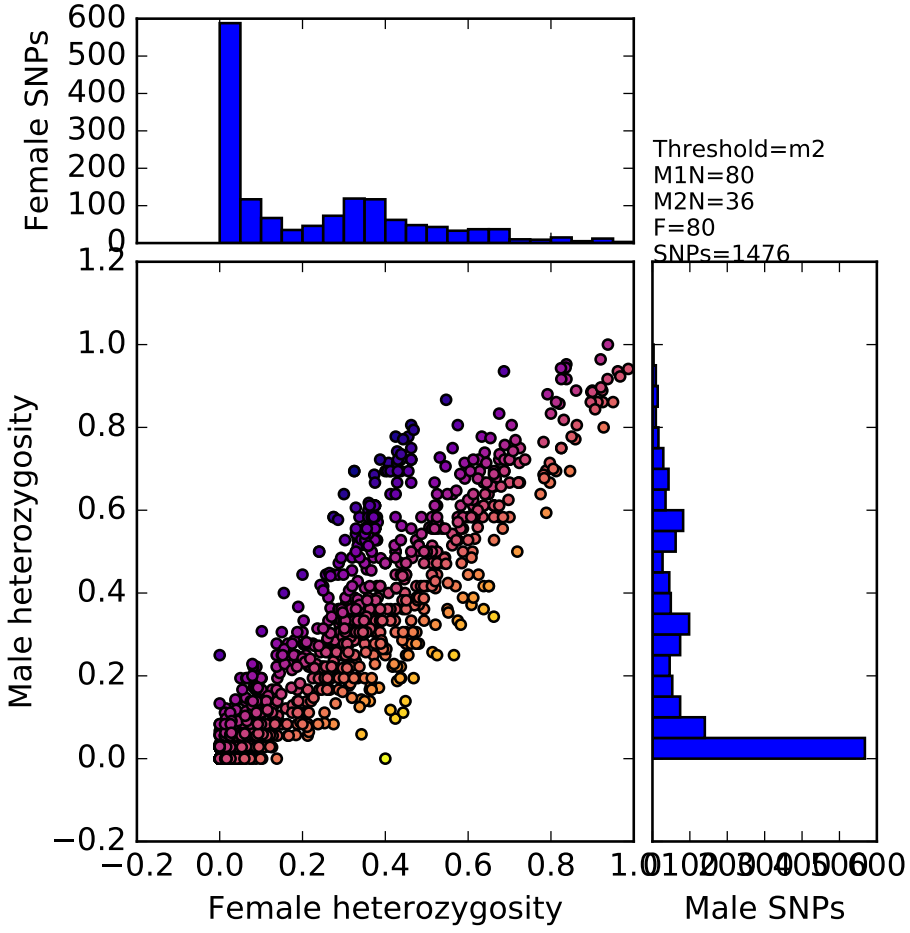

Figure 4.2: Homozygosity of female and diploid male SNPs where depth ≥ 5 (permissive parameters). Each point on the scatterplot is a SNP, and its coordinate are the proportion of females (x) and males (y) in which it is homozygous. Histograms allow to visualize the distribution of homozygosity for SNPs of each sex. The color code shows how SNPs fit the CSD pattern with lighter points being closer to it (i.e. more homozygous in males and heterozygous in females). Summary statistics on the top right show the threshold used, the number of haploid males excluded (M1N) as well as the number of diploid males (M2N), females (F) and SNPs included.

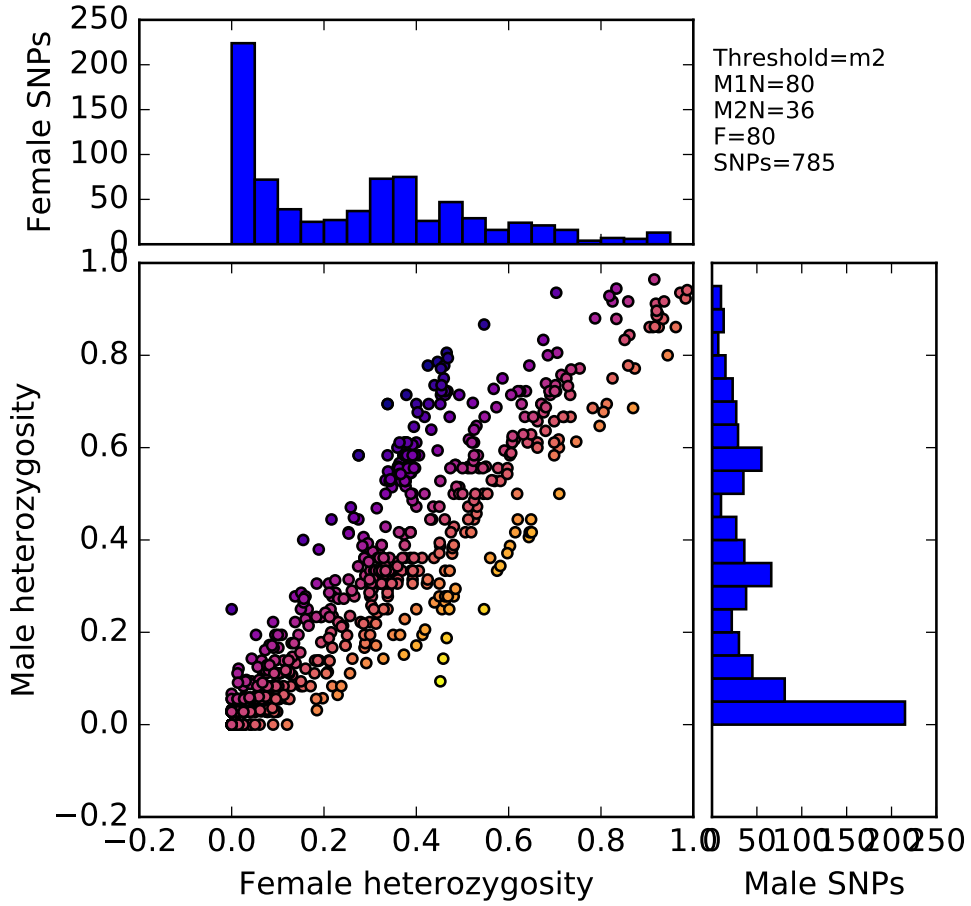

Figure 4.3: Homozygosity of female and diploid male SNPs $\geq$ 25 (stringent parameters). Each point on the scatterplot is a SNP, and its coordinate are the proportion of females (x) and males (y) in which it is homozygous. Histograms allow to visualize the distribution of homozygosity for SNPs of each sex. The color code shows how SNPs fit the CSD pattern with lighter points being closer to it (i.e. more homozygous in males and heterozygous in females). Summary statistics on the top right show the threshold used, the number of haploid males excluded (M1N) as well as the number of diploid males (M2N), females (F) and SNPs included.

Chapter 5

Cleaning genomic data

date : 23.06.2017

The first attempt at separating individuals based on inbreeding coefficient revealed several issues inherent to the data:

- The inbreeding coefficient is not clearly bimodal in all families (1 haploid mode and 1 diploid mode)
- Some loci have more than 2 haplotypes, although we are working with diploids.
- There are no loci that clearly stand out as perfectly 'CSD like' among all individuals, this should be looked at family-wise.

These issues will be solved one by one in this chapter.

5.1 Improving ploidy separation metric

The inbreeding coefficient was computed using all males/females. Also, the raw heterozygosity may be a more accurate metric than inbreeding coefficient. Both issue require the analysis to be done in a per-family fashion, either by re-running the populations program for each family, or implementing family as an additional population layer.
Date: 27.06.2017
Solution: I could switch to observed homozygosity, but I think it is also fine to keep this metric (I can always change it later). I did however, re-run the populations program separately for each family. I did not add a population layer to the popmap, because it would not have allowed for different loci blacklists between families when solving the third issue. Visualizing the distribution of inbreeding coefficients across individuals reveals a pretty good (?) separation of individuals (Figure 5.1).

Date: 12.07.2017 Update: This method of ploidy separation is flawed as it will overestimate the number of haploid males in families with few females. Indeed, with a lower standard deviation of female homozygosity, the threshold tends to the mean only.

Because the homozygosity of diploid offspring depends on its mother, but that of haploids are independent of mother background, I can probably use a universal threshold for all families. This way, the threshold can be equally conservative in all families and not biased towards certain of them. Plotting homozygosity of all individuals (Figure 5.2) revealed 2 separate meta-distributtions. The one on the left is a gaussian made up of several smaller ones (each smaller distribution is a family, with the mean depending on the mother), whereas the distribution on the right is made of a single gaussian containing all (or most) haploid males. Choosing a fixed conservative threshold in between should allow to keep more individuals and remove bias.

5.2 Loci with 3 or more haplotypes

There are two possible explanation for the existence of these loci. It could be either due to a contamination, in which case few individuals would present many loci with 3 or more haplotype. On the other hand, if few

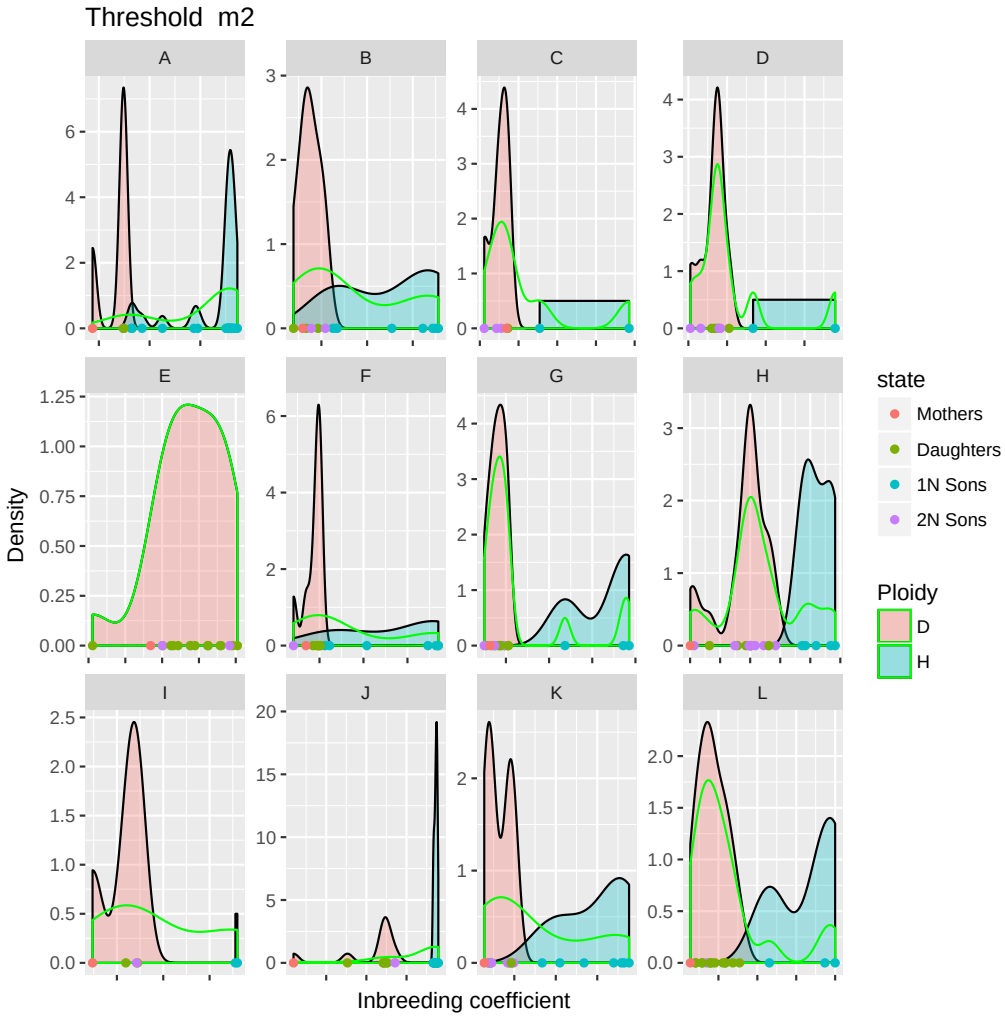

Figure 5.1: Distriution of individual inbreeding coefficient. The green curve represents the overall distribution of inbreeding coefficients within the family. The blue and pink areas represent the distribution of inbreeding coefficients for haploids and diploids, respectively, split using the m2 threshold defined at the beginning of chapter 4. Ideally, the green curve would follow a bimodal distribution clearly separating individuals by homozygosity.

loci have this issue in several individuals, it could be due to paralog merging. This can be solved by identifying which case is the most likely, and either excluding concerned individuals (first case), or loci (second case).

24.06.2017: Asked on STACKS google group about the haplotypes.tsv file. It is apparently normal to have more than 2 genotypes and depends on the parameters that were set in pstacks/ustacks. The answer I got, is that I should only work with the VCF file, which calls only 2 genotypes/site/individual. This imply figures 4.3 and 4.2 are not valid, as I misinterpreted the file content. I will re-generate these figures using the VCF file directly.

Date: 27.06.2017

Solution: I used the -012 argument in VCFtools to produce a genotype matrix from the populations output VCF file. This matrix encodes the genotype of each individual at each SNP with an integer representing the number of non-reference alleles present. These matrices were generated separately for each family and used to regenerate the figures 4.3 and 4.2.

The issue when visualizing the SNPs heterozygosity per family (Figure 5.3) is that the low number of individuals allow for few different heterozygosity values for any given SNPs, yielding a high number of 'perfectly' CSD-like candidates. Solving the next issue or grouping SNPs by contig/locus may help reduce this number, but ultimately only association mapping will allow to find the best candidate region.

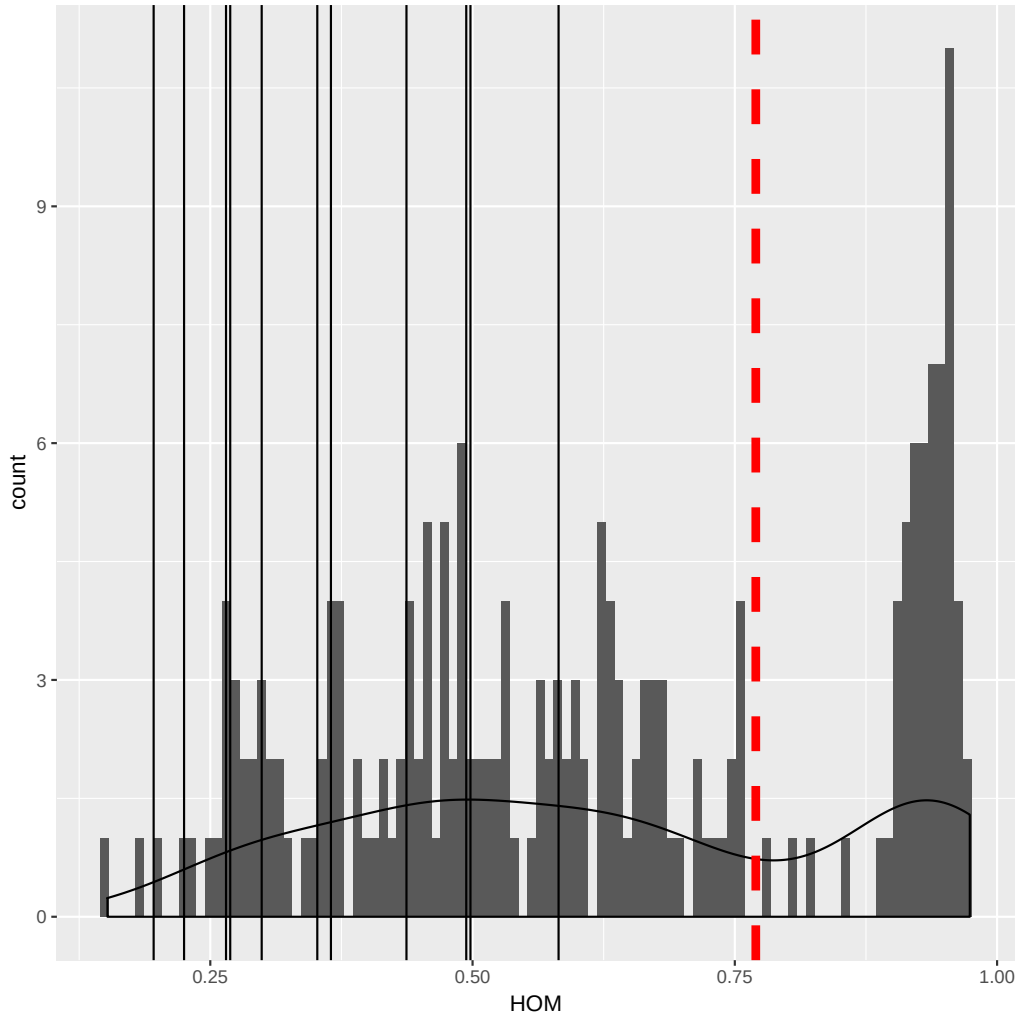

Figure 5.2: Distribution of the proportion of homozygous loci across all individuals. Continuous vertical black lines are the mothers values and the dashed vertical red line is the (visually) optimal threshold, fixed at 0.77.

5.3 No clear signal in whole population

This issue could be expected from the biological system because it is believed that there is ml-CSD in this species; there should be no single locus that will fit the CSD pattern across all individuals. This mean we need to identify different loci independently in each family. There are several things which can be done to work around this issue.

First, and most importantly, the populations program should be taking family information into account, but this was done already when solving the first issue (Improving ploidy separation metric). Second, the data should be cleaned by blacklisting loci that are homozygous in mothers in each family, as there is no way these will be heterozygous in their respective daughters and be CSD candidates.

Date: 29.06.2017

Solution: I re-ran the explo assoc script used to generate figure 5.3 and excluded all SNPs that are homozygous in the mother (Figure 5.4). It removed a significant number of SNPs from the set, but there are still way too many that fit the CSD pattern.

family	N SNPs
A	167
B	77
C	61
E	166
F	158
G	345
H	230
I	32
J	193
K	40
L	128

Table 5.1: Number of homozygous SNPs found in the mother of each family. Note family D is excluded since there was no genomic data available for this mother.

family	N SNPs
A	15
B	38
C	45
D	184
E	47
F	199
G	62
H	62
I	29
J	144
K	112
L	37

Table 5.2: Number of homozygous SNPs that fit the CSD pattern in each family.

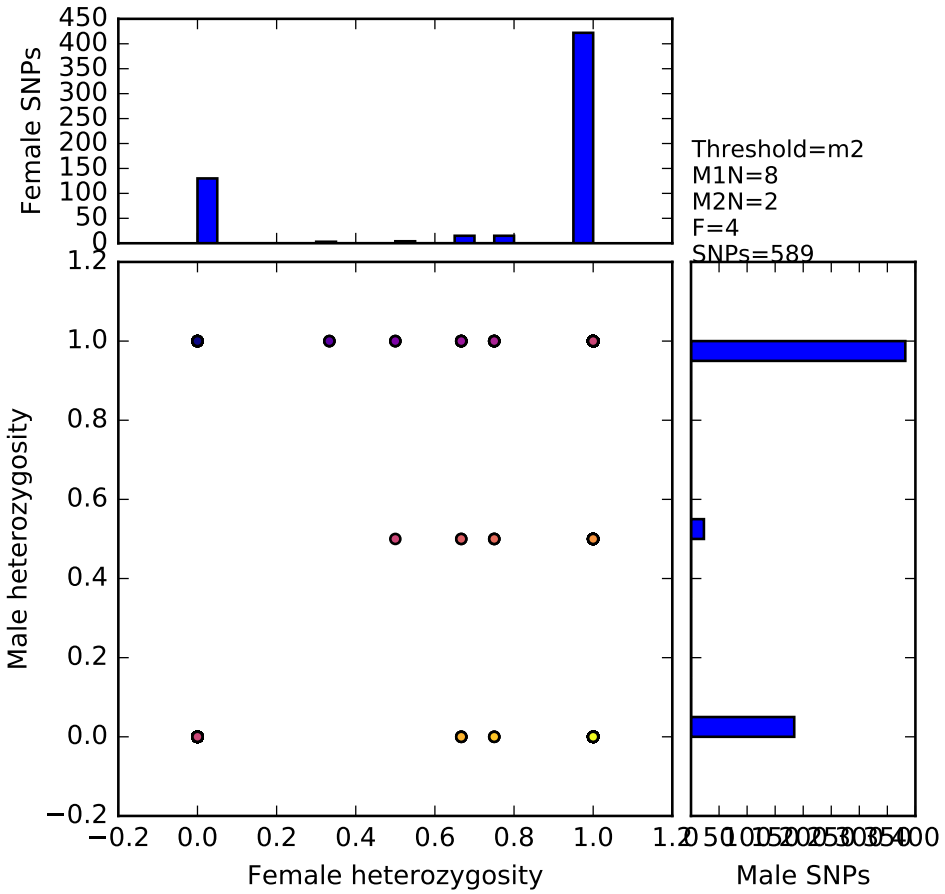

Figure 5.3: Example: SNPs heterozygosity across individuals of family B. The yellowness represents the proximity to CSD-like pattern (i.e. homozygous in all males and heterozygous in all females. Note: populations parameters set to : min depth (D)=5 and prop pop (r)=75

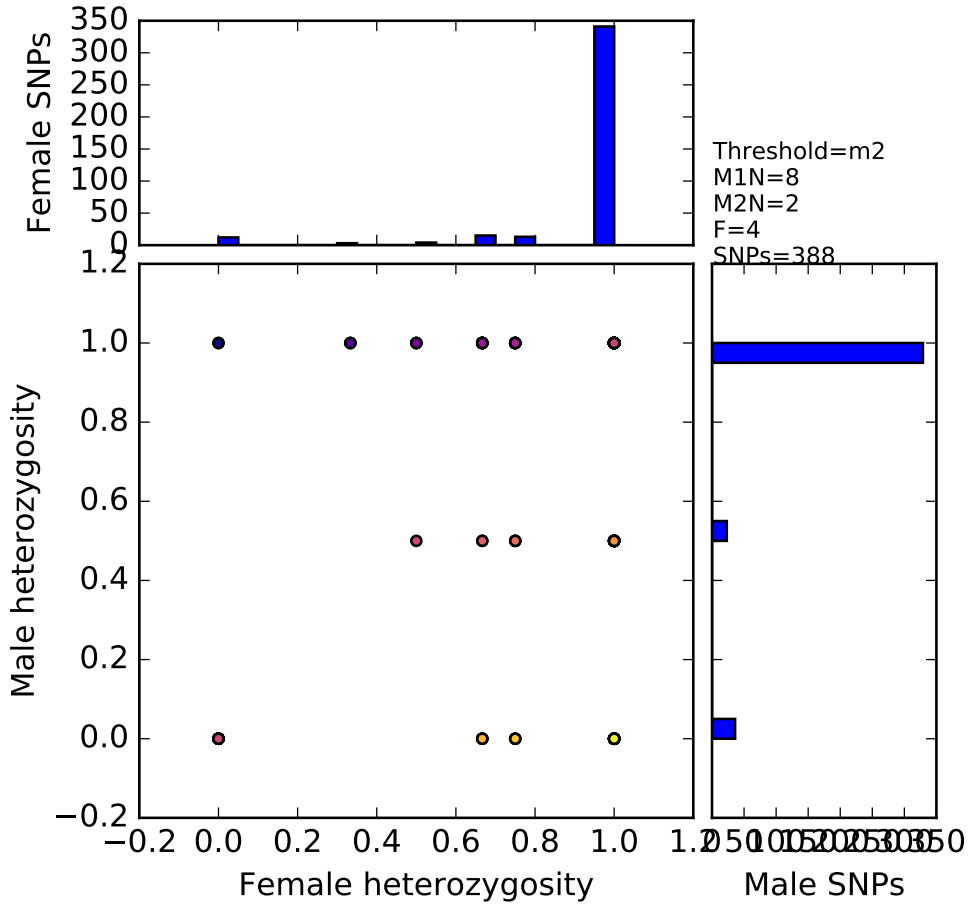

Figure 5.4: Example: SNPs heterozygosity across individuals of family B after removing SNPs that are homozygous in the mother. The yellowness represents the proximity to CSD-like pattern (i.e. homozygous in all males and heterozygous in all females. Note: populations parameters set to : min depth (D)=5 and prop pop (r)=75

Date: 24.07.2017
I removed loci that are homozygous in mother using a second technique:

- For each mother: use Pstacks snps file to find sample ID of all loci for which all position are "O" (for homozygous).
- Look up the sstacks matches file of each mother to retrieve the matching catalog ID
- exclude corresponding catalog ID for the family in populations output files in downstream analyses.

I am still not sure if this technique is safe / correct, the steps described above are currently implemented in the hom_filt function of Hom_M-F.R.
Date: 28.07.2017
I am now using a much simpler method which should be relatively correct. I use the output populations file batch_0.sumstats.tsv to remove the SNPs that have no alternative nucleotide in the female population. I am therefore assuming that if the mother is homozygous, all daughters are as well. That means I might keep some SNPs that are in fact homozygous in the mother, either due to sequencing errors or point mutations, but these events should be extremely rare.

5.3.1 Transmission bias

Here I check how frequently SNPs that are homozygous in a mother are heterozygous in its offspring and vice versa. (to be continued)

Chapter 6

Ordered genome

Date: 05.07.2017

Now that the data is cleaner, I will start looking for CSD. Casper provided me the genome with the contigs ordered by chromosomes (but not necessarily oriented correctly) he obtained with his linkage map. I will use this as the new reference genome and run the pipeline all the way from the mapping with it.

Date : 08.07.2017

I ran the whole pipeline on the ordered genome. Here are all statistics associated with all steps of the analysis previously described.

Statistics	Values
Assembly length (Mbp)	140.7
Largest scaffold (Mbp)	17.2
Mean scaffold size (kbp)	100
Median scaffold size (kbp)	25.7
N50 (kbp)	9518.5
Number of scaffolds	1408

Table 6.1: Assembly statistics for the ordered reference genome of *Lysiphlebus fabarum*

	tot	single	multi	miss
Number	170435458	94668507	44531856	31235095
proportion	1	0.555	0.261	0.183

Table 6.2: Mapping: Proportion and number of reads aligned on the ordered reference genome using BWA's aln algorithm with 4 mismatches allowed.

min˙cov	nloci	mean˙cov	sd˙cov
3	2694.0	50.0	69.4

Table 6.3: Pstacks: Summary statistics of stacks obtained with minimum coverage set to 3 in pstacks, using the ordered genome.

mismatch	mean˙loci	mean˙alleles
3	9072	34311

Table 6.4: Cstacks: Summary statistics of loci obtained with a mismatch value set to 3 in cstacks using the ordered genome.

Chapter 7

Number of CSD loci

It is possible to approximate the number of CSD loci heterozygous in a mother using the proportion of males among diploid individuals. Provided the loci are on different chromosomes, if n CSD loci are heterozygous in the mother, there should be $\frac{1}{2^n}$ of males among diploid, because as long as heterozygosity is retained at one of these loci, the offspring should be female.

7.1 Categorizing mothers

The mothers can be categorized based on this proportion of male offspring. The categories will reflect different combination of heterozygous CSD loci. Assuming the recombination rate is the same in different mothers, the different categories will depend only on the distance of the heterozygous locus/loci from the centromere. Loci further from the centromere will be undergo gene conversion more frequently.

There are different scenarii:

- 1 CSD locus: In this scenario, all mothers will be heterozygous at this locus and there should only be one category since the proportion of males among diploids will only depend on the distance of this locus from the centromere.
- 2 CSD loci, same chromosomal arm: In this scenario, there should be a single category as only a recombination proximal to both loci would generat diploid males, therefore diploid male production only depends on the recombination rate of the locus that is closest to the centromere.
- 2 CSD loci, different chromosomal arms: This scenario would allow for 3 categories of mothers; those heterozygous at one, the other, or both CSD loci. Those heterozygous at the furthest locus only should have the highest production of diploid males as recombinations will be more frequent. Those with only the closest one would have a lower production. The females heterozygous at both loci would have the lowest production of diploid males (proportion ofshould be the product of the two other categories).
- 3 or more CSD loci: In case there are more than 2 loci, the number of categories would quickly increase beyond the scope of this study as we would not have enough families to detect the categories.

Note that scenario 2 is only happening if both loci are on the same side of the centromere, otherwise two separate recombination events would be required to cause male development.

Update: 02.08.2017

I need more families to build solid categories. The 12 families currently available seem to form distinct clusters, but more are needed to ensure a solid classification. Another issue is that we are not sequencing all offspring (for financial reasons) and we select the relative number of males and females to increase the power of all analysis (i.e. trying to balance the numbers of each). This will bias the numbers and we should account for it.

To address the latter problem, I propose the following strategy: If m out of M males and f out of F females are selected for sequencing, compute the proportion of diploid d among the m males. Then, compute the proportion of males p among total offspring as $p = \frac{M*d}{F+M}$, thus extrapolating the proportion of diploid males in the family to non-sequenced samples. This should not be biased as we select them irrespectively of ploidy.

However, families with small number of sequenced males are vulnerable to randomness and extrapolating might be dangerous in these.

To solve the issue of low number of families, we are preparing new libraries for sequencing, both expanding existing families and adding new ones. In total, we will add more than 200 individuals, but the exact number is subject to change.

Once the mothers have been categorized based on the proportion of males in their diploid offspring, I will use this category information to filter CSD candidates common to mothers within one category. Assuming scenario 3, for instance, there will be 3 categories. I will first filter the best hits common to the category with the lowest production of diploid males. I will then look for overlap with the two other categories separately.

Date: 24.08.2017

I used k-means clustering on proportion of males among diploid offspring to categorize families. I assumed 3 clusters

Date: 25.08.2017

There may be an issue with the use of proportion of 2n males to categorize mothers; when we select individuals for sequencing, we try to balance the proportion of males and females to make sure we have enough of both sex. This will likely decrease the proportion of diploid males in families with many males as we do not use all of them. We could solve this by using the proportion of males in all offspring instead, assuming the proportion of haploidy is the same in all families. This way, we could use the total sex ratio of the family (including non-sequenced individuals) to categorize mothers. I should ask Casper if he has this information and the opinion of Tanja on that matter.

Date: 08.09.2017

I now have the information of the total number of offspring for each sex per family and used it to categorize mothers. I multiplied the rate of haploidy in sequenced males by the total number of males in each family to infer the total number of diploid males. I then divided that number by the total number of offspring in each family to classify mothers. I am currently using families from libraries 6, 7, 7b and 10b which sums up to a total of 21 families. If I consider a 2 CSD loci case with 3 categories of mothers (Figure 7.1), I do not obtain clear cluster. If I consider a 3 CSD loci case with 7 categories, I do not have enough families to establish groups (Figure 7.2), with the current libraries.

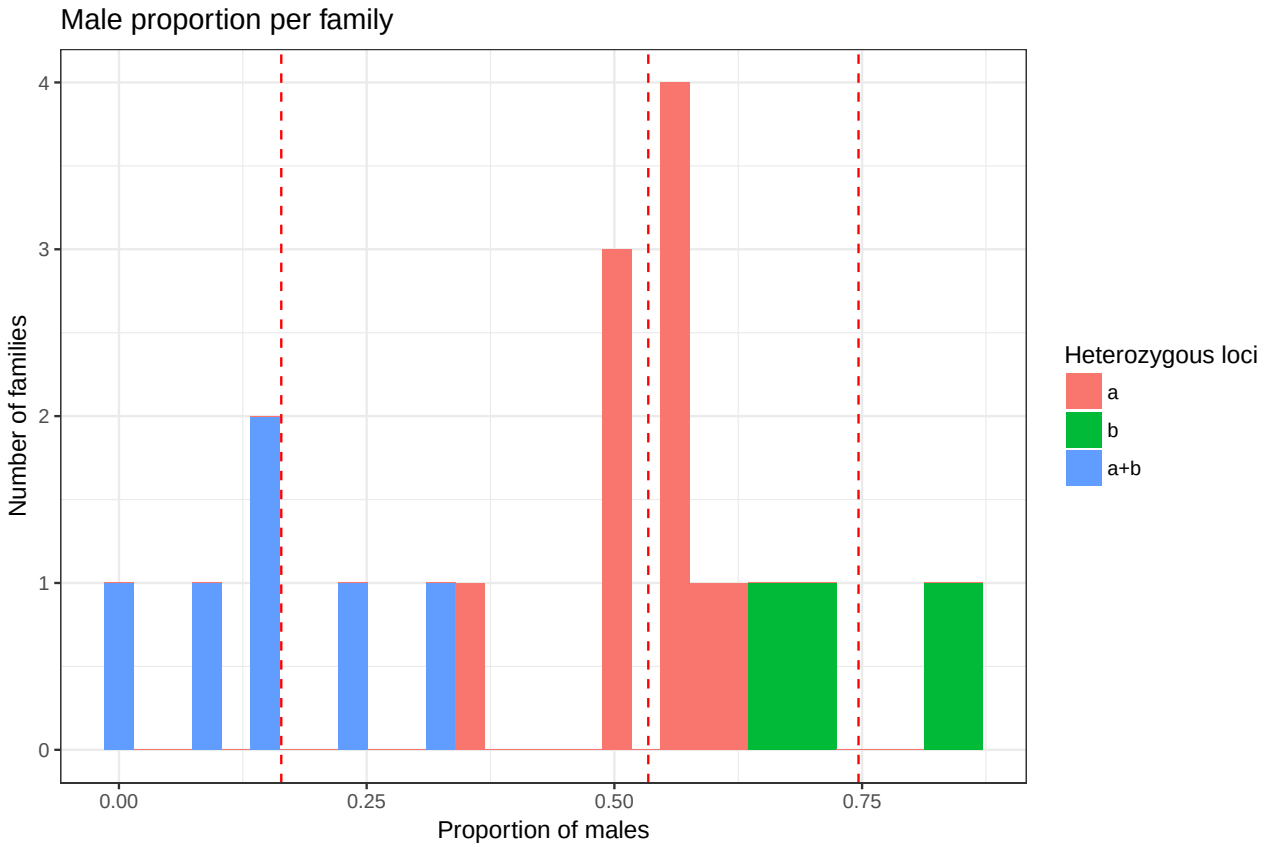

Figure 7.1: Distribution of families according to inferred proportion of males among diploid offspring considering 2 CSD loci. Families are separated into 3 groups depending on which loci are expected to be heterozygous in the mother. Families were grouped using 1D k-means clutering and vertial dashed lines represent the center of each cluster.

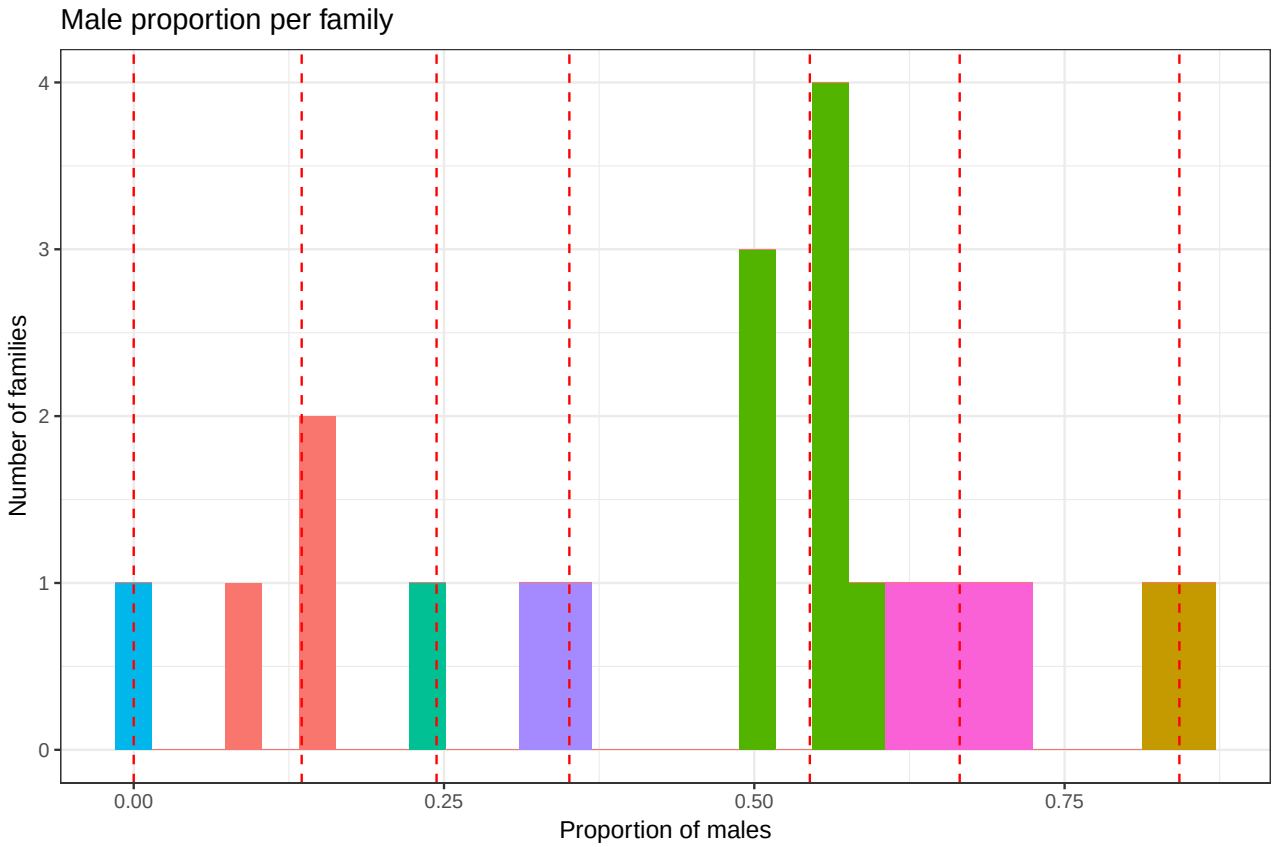

Figure 7.2: Distribution of families according to inferred proportion of males among diploid offspring considering 3 CSD loci. Families are separated into 7 groups depending on which loci are expected to be heterozygous in the mother. Families were grouped using 1D k-means clutering and vertial dashed lines represent the center of each cluster.

7.2 Distance to centromeres

To validate hits, I will use the information of the distance from centromere. The position of the centromere in the scaffold (i.e. metacentric vs telocentric chromosome) can be inferred from the recombination rates along the chromosome. The method I am going to use for this is to measure the proportion of offspring homozygous at each SNP that is heterozygous in the mother (i.e. proportion of recombinant offspring at that SNP).The centromere should be the region with the lowest value. The hits obtained for CSD candidates in the different categories should be coherent with this distance; the hit in the family with a low diploid male production should be closer to the centromere.

Modelling the recombination rate along chromosomes using a weighted loess (lowess) curve revealed local minima that could contain centromeres, however the position of these minima can vary strongly depending on the span (smoothing parameter) allowed for local regression when building the curve. This is likely due to the small number of individuals, or rarity of recombination events. The proportion of homozygosity also tends to decrease far from centromere, this is likely due to multiple recombination events restoring heterozygosity. Note the curve uses the proportion of recombinant offspring within a family at a given SNP weighted by the total number of offspring in that family.

I tried locating the centromere both by running populations with all individuals pooled together (Figure 7.3) and separately for each family (Figure 7.4). If I want to use the minimum estimate of local regressions objectively to locate the centromere, I will need to use some form of cross validation to estimate the span. I could also use a visually optimal value for span; the default value used in ggplot's geom_smooth() is 0.75 and seems reasonable on my data (does not seem too noisy).

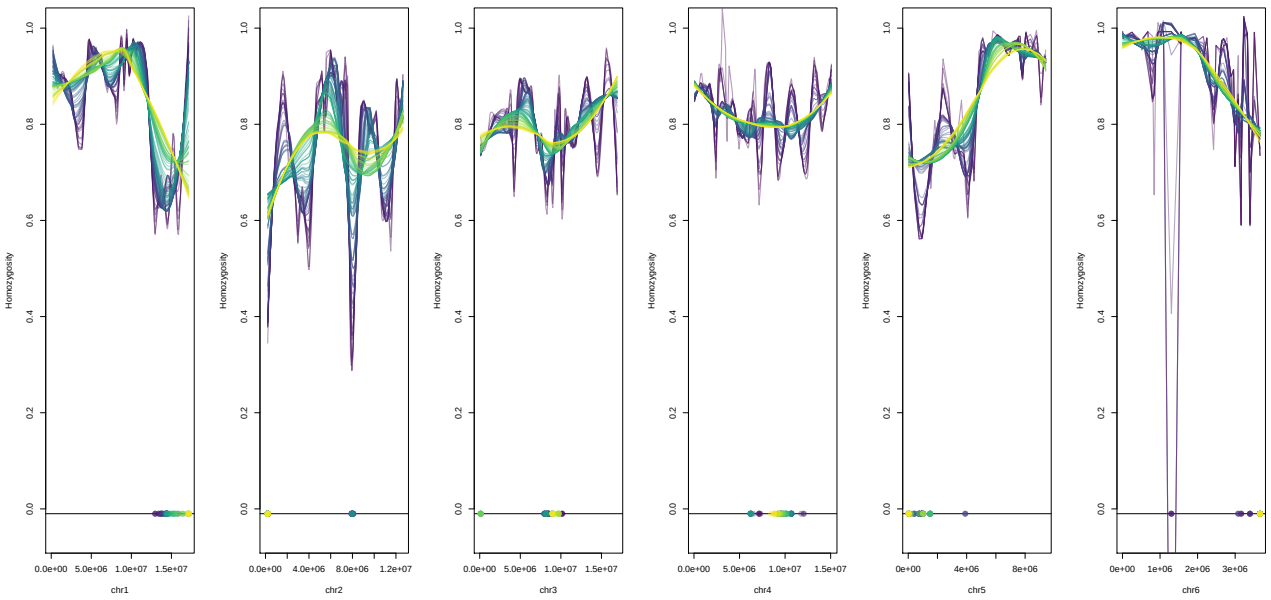

Figure 7.3: Modelling recombination rates along chromosomes using local regression with second degree polynomials on proportion of homozygous individuals as a proxy. Populations was run on all individuals pooled together (r80, mindepth:3). Colors represent different span value for the local regression and colored dots are the inferred centromere position (i.e. the minumum of the curve) with the corresponding span value.

Date: 12.08.2017

To obtain better result when trying to locate centromeres, I used the genomic output from populations to include fixed SNPs. I subsequently removed all SNPs that are **either missing or homozygous** in the mother from its whole family (including itself). The variation induced by the span parameter when using local regression is now much smaller. To make sure the centromeres are called correctly, I tried a second simpler

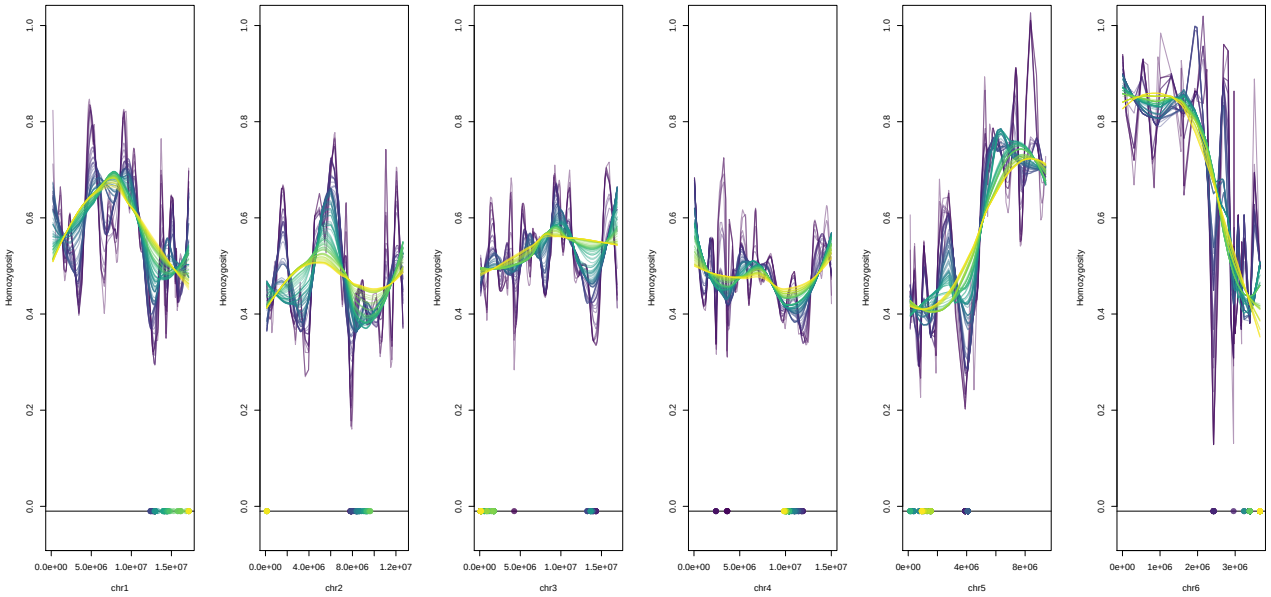

Figure 7.4: Modelling recombination rates along chromosomes using local regression with second degree polynomials on proportion of homozygous individuals as a proxy. Populations was run on separately for each family, SNPs where the mother is homozygous were removed in each family (r80, mindepth: 3).Colors represent different span value for the local regression and colored dots are the inferred centromere position with the corresponding span value.Colors represent different span value for the local regression and colored dots are the inferred centromeres position (i.e. the minumum of the curve) with the corresponding span value.

method, which consists in computing means over a sliding window on each chromosome. I tried both methods with several different parameter values (Figure 7.6). The methods yielded similar results and I visually selected reasonable parameter values for span and window size to compare the centromeres positions inferred by both methods (Figure **??**). Centromeres are merely regions and have no objective precise genomic position, besides I will only be interested in relative differences in distance from centromeres to select loci, therefore the parameter values I chose will probably have little to no effect on the results.

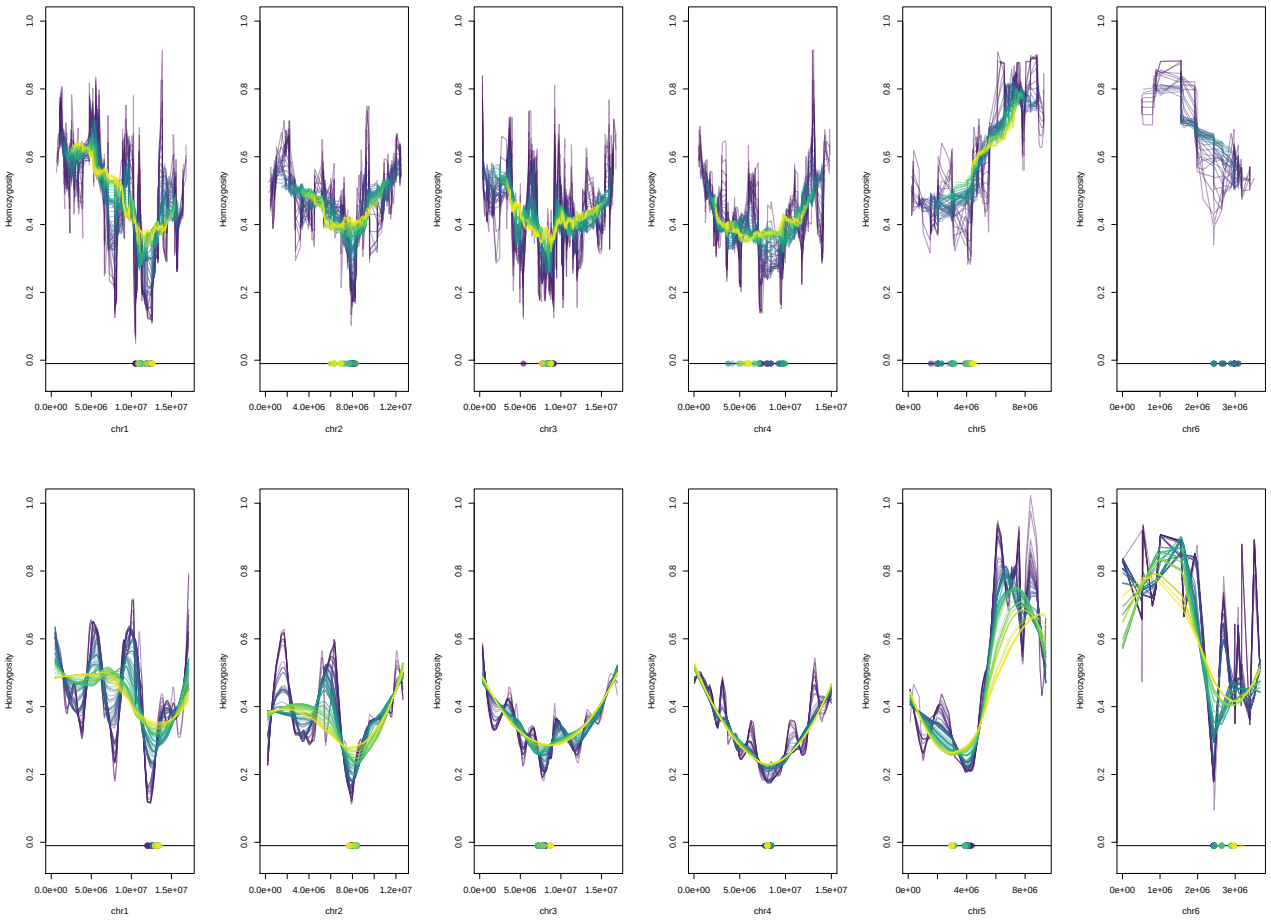

Figure 7.5: Comparing moving averages and local regression with second degree polynomials to approximate recombination rates. Populations was run on all individuals pooled together. Fixed SNPs are included and SNPs where the mother is homozygous or that were absent in the mother were removed in each family (r80, mindepth: 3). Range of parameter tested: span varied between 15% and 100% of observations by intervals of 1% for local regression and window size between 3 and 80 observations with intevals of 1 for moving averages. Colors represent the different span or window size value for the local regression and colored dots are the inferred centromeres position (i.e. the minimum of the curve) with the corresponding parameter value.

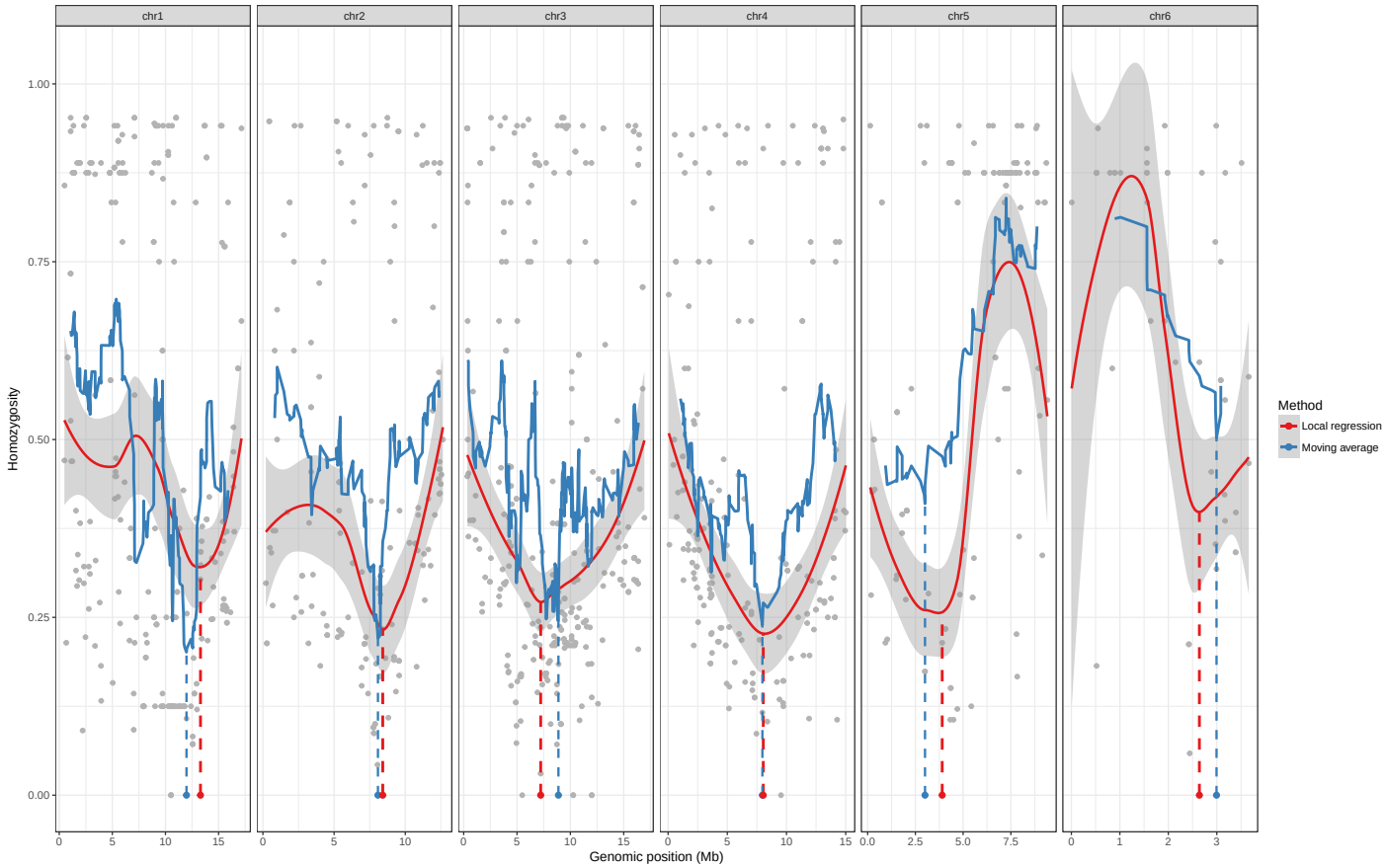

Figure 7.6: Comparing moving averages and local regression with second degree polynomials to approximate recombination rates. Populations was run on all individuals pooled together. Fixed SNPs are included and SNPs where the mother is homozygous or that were absent in the mother were removed in each family (r80, mindepth: 3). Parameters selected for comparison: window size: 20, span: 0.75. Dotted lines show the inferred centromeres position (i.e. the minimum of the curve) with both methods.

7.2.1 Terminal or central fusion

Date: 31.08.2017

When modeling heterozygosity along the genome, there were interferences (waves) and the putative centromeric regions are not fully heterozygous. This could be due to several factors (inclusion of haploids, sequencing errors, errors in linkage map...). One of the more likely reasons, could be that some individuals are produced by terminal fusion automixis, rather than central fusion, which would cause them to be homozygous at the centromere and affect the whole model.

A way to test this, which has already been used in Daphnia (Svendsen et al, 2015), is to model the heterozygosity relative to the mother of each offspring as a function of the distance from the centromere. All chromosomes are pulled together. Note the Svendsen uses centimorgans and I use physical distance in basepair.

I visualise it in two ways:
First, using a sliding window moving away from the centromere in absolute distance (Figure 7.7). Each sliding window computes the proportion of heterozygous sites in each individual.
Second, choosing different sizes for the putative centromeric region and computing the proportion of heterozygous with each value (Figure 7.8). Under central fusion, one would expect the heterozyosity to decrease when the window moves away from the centromere, or when the putative region is enlarged. On the other hand, under terminal fusion, the value should increase when moving away or increasing the size of the centromeric region.

Given the density of SNPs in my data, these methods cannot be used to reliably identify central or terminal fusion. There is also the possibility that some scaffolds are inverted in the linkage map, which would strongly affect the results.

tl;dr : I don't think I can pull anything useful from this, data is too noisy.

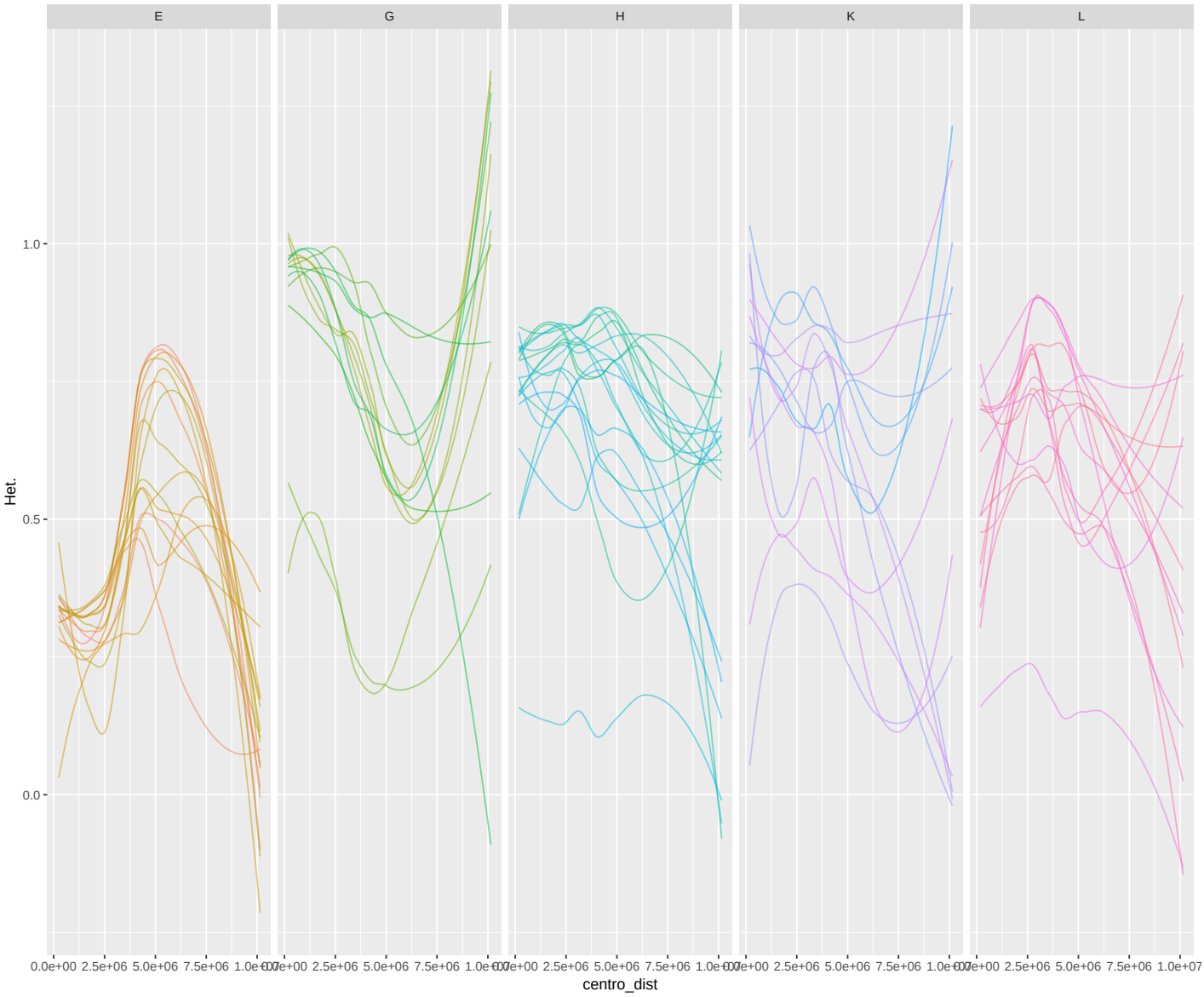

Figure 7.7: Modeling the proportion of heterozygosity as a function of the distance from the centromere in each individual separately. Proportions are calculated in sliding windows, each containing 30 SNPs and the base pair at the middle of the window is used for plotting. The curves are obtained by local regression (loess). All chromosomes are pulled together in a single figure

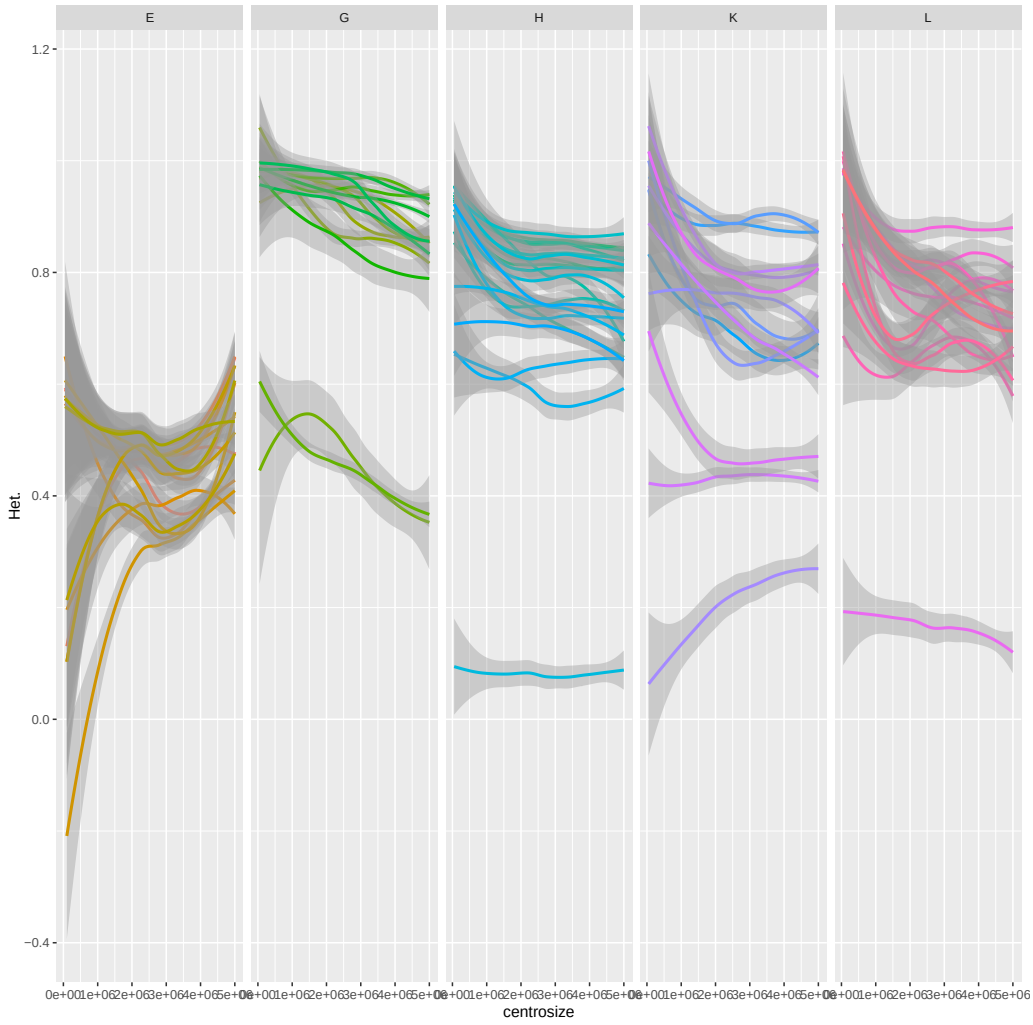

Figure 7.8: Measuring the proportion of heterozygous sites relative to mother in regions of variying size between +/- 50kb and +/- 5Mb around centromere in each individual. Panes represent families (only 5 largest ones included) and chromosomes are pooled together. The curves are obtained using a local regression (loess).

Update: 13.09.2017

To differentiate more reliably central vs terminal fusion individuals, I used a simpler method (Figure 7.9). After transforming basepair coordinates in each chromosome into absolute distance from centromere, I separate each chromosome into 2 windows: "centro" for centromeric region and "telo" for telomeric region. The centromeric region contains all SNP located closest to the centromere within 25% of the maximum distance to centromere recorded in the chromosome, while the telomeric region contains those in the furthest 25%. I compared the proportion of heterozygous SNPs in the two windows, first separating chromosomes (Figure 7.10). Then, I tried pooling all chromosomes together in each individual and I compared the two windows using a linear model, weighting the value of each chromosome by the number of SNPs contained in the window (Figure 7.11).

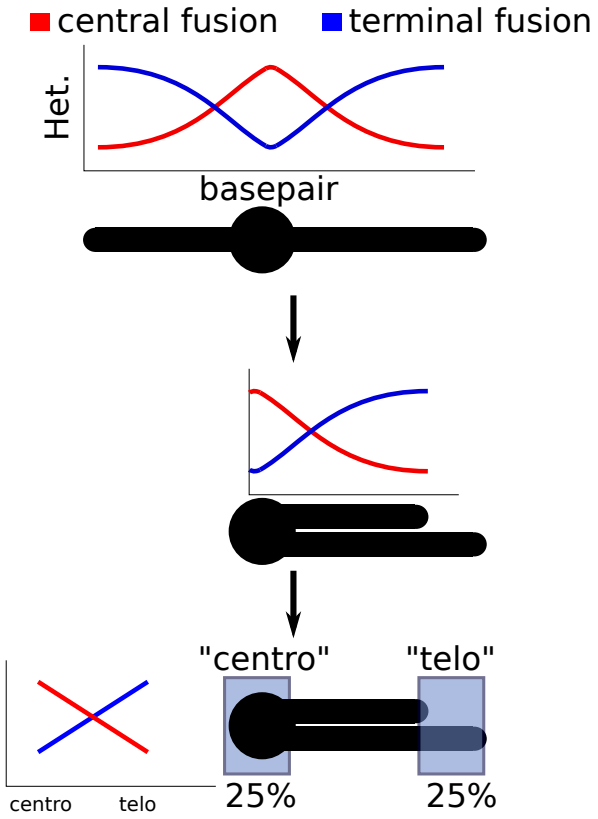

Figure 7.9: Method used to identify central fusion or terminal fusion individuals from genome-wide heterozygosity. Central fusion individuals (red) are expected to have high heterozygosity values decreasing towards centromeres while terminal fusion (blue) individuals are expected to have low heterozygosity increasing towards centromeres. The basepair coordinates of SNPs are first transformed into absolute distance from centromere. SNPs closer to the centromere than 25% of the maximum distance to centromere in each chromosome are pooled into one "centromeric" window and those in the 25% closest to the telomeres are pooled into another "telomeric" window. The proportion of heterozygous SNPs is then computed separately in each window and the difference between those values is used to assign fusion mechanism.

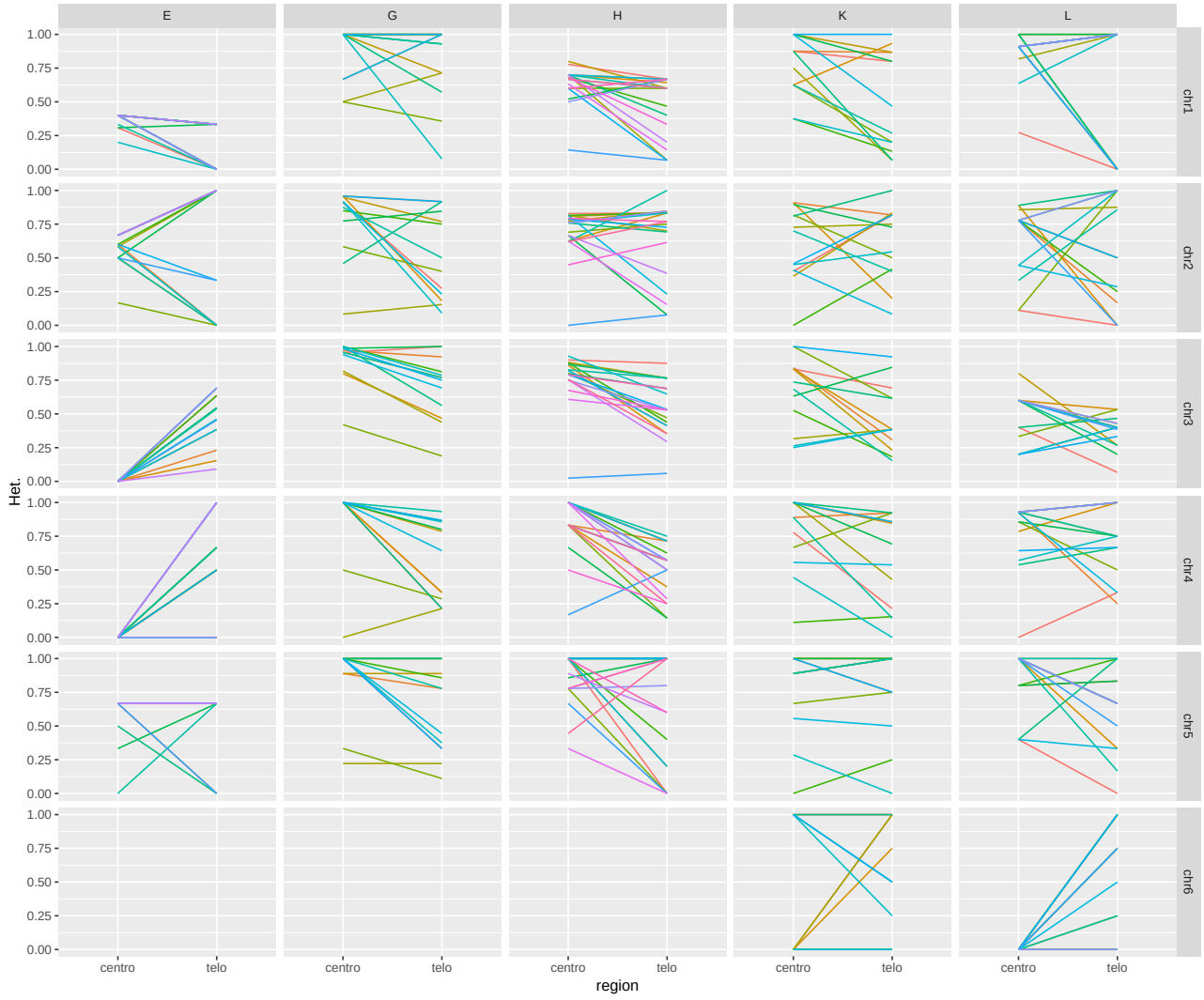

Figure 7.10: Comparing the proportion of heterozygous SNPs in each chromosome among variant sites between the centromeric window (SNPs closer than 25% of maximum distance to centromere) and telomeric window (SNPs further than 75%). Only the 6 largest families are included for visualisation purpose.

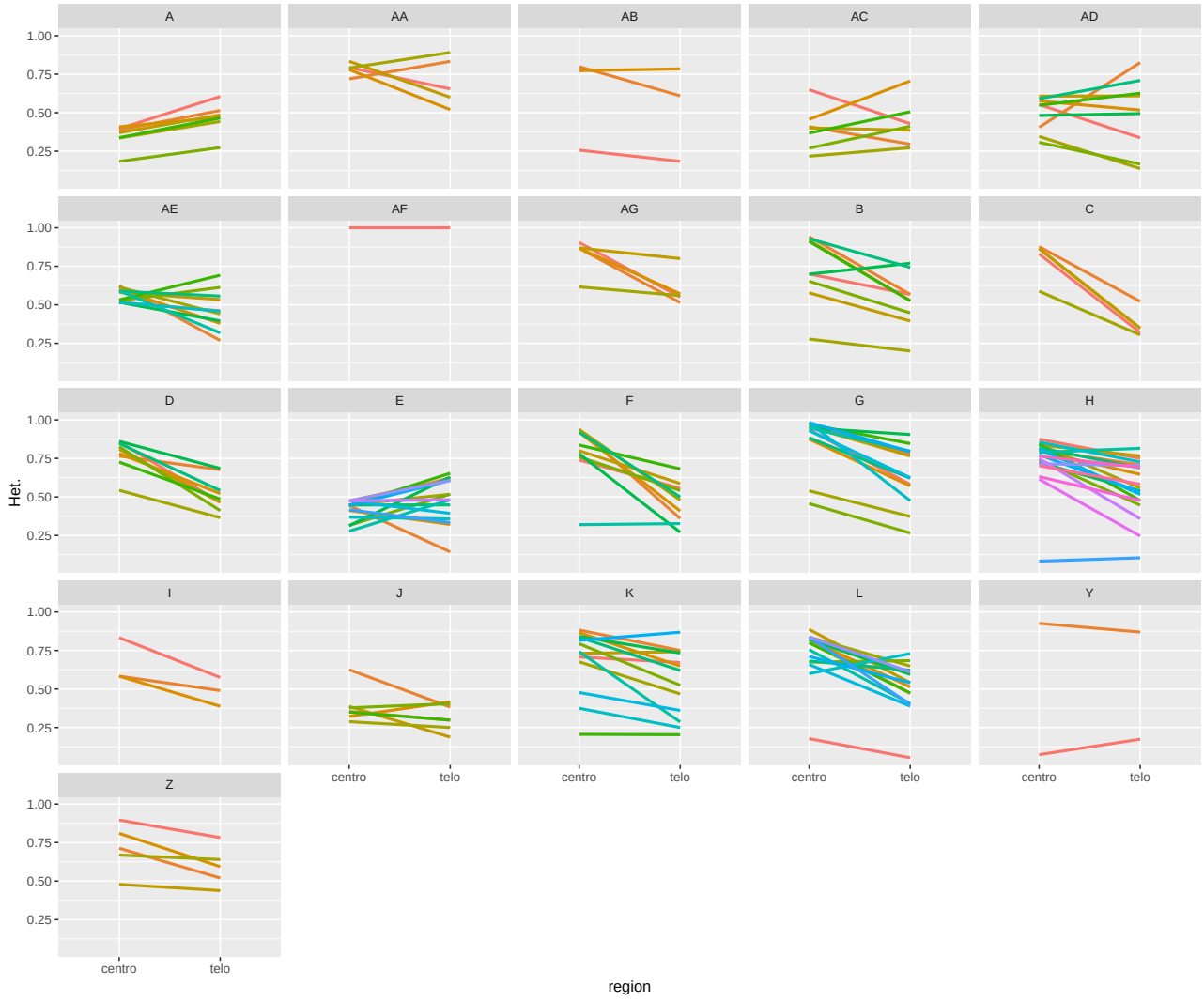

Figure 7.11: Comparing the proportion of heterozygous SNPs pooling all chromosomes together, between the centromeric window (SNPs closer than 25% of maximum distance to centromere) and telomeric window (SNPs further than 75%). A linear model is performed to compare the 2 windows and the value of each chromosome is weighted by the number of SNPs used to compute the window.

Date: 14.09.2017 A more robust way of separating CFA from TFA individuals would be to only use the centromeric window and rely on the distribution of heterozygosity within that window to classify individuals. Just as above, I used a window including all SNPs around the centromere located within a region of 25% of the maximum distance from the centromere within each chromosome. I first looked at the distribution separately for each chromosome (Figure 7.12) and by pooling all chromosome together (Figure 7.13, using a weighted mean with the number of loci in the windows as the weights.

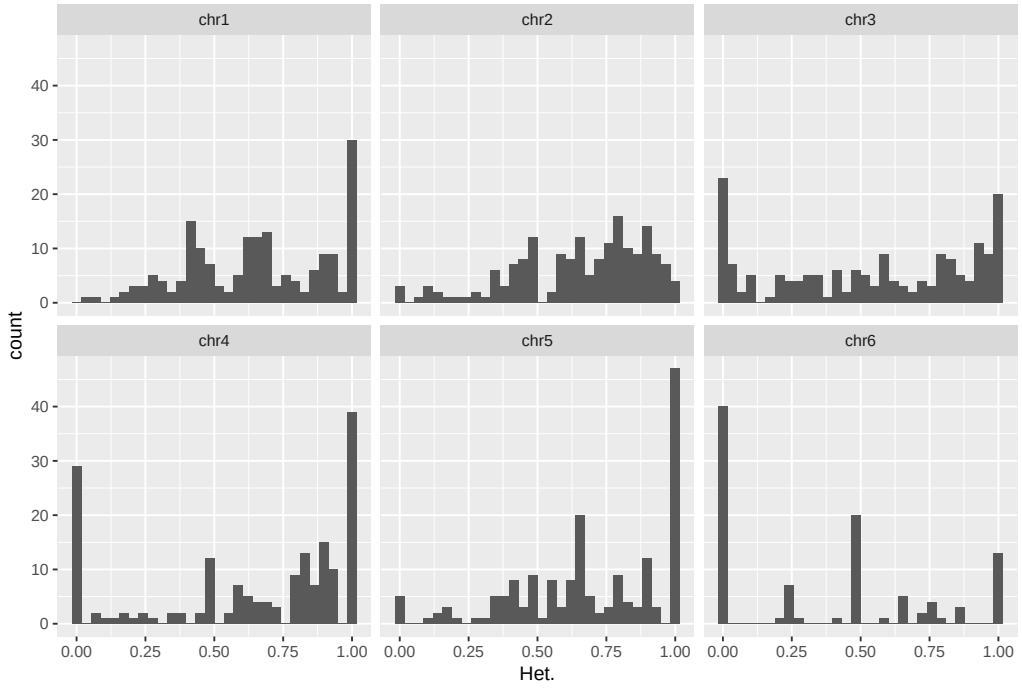

Figure 7.12: Distribution of heterozygosity in centromeric windows across all individuals from libraries 7, 7b and 10b. Centromeric windows contain all SNPs located within 25% of the maximum distance to centromere in each chromosome.

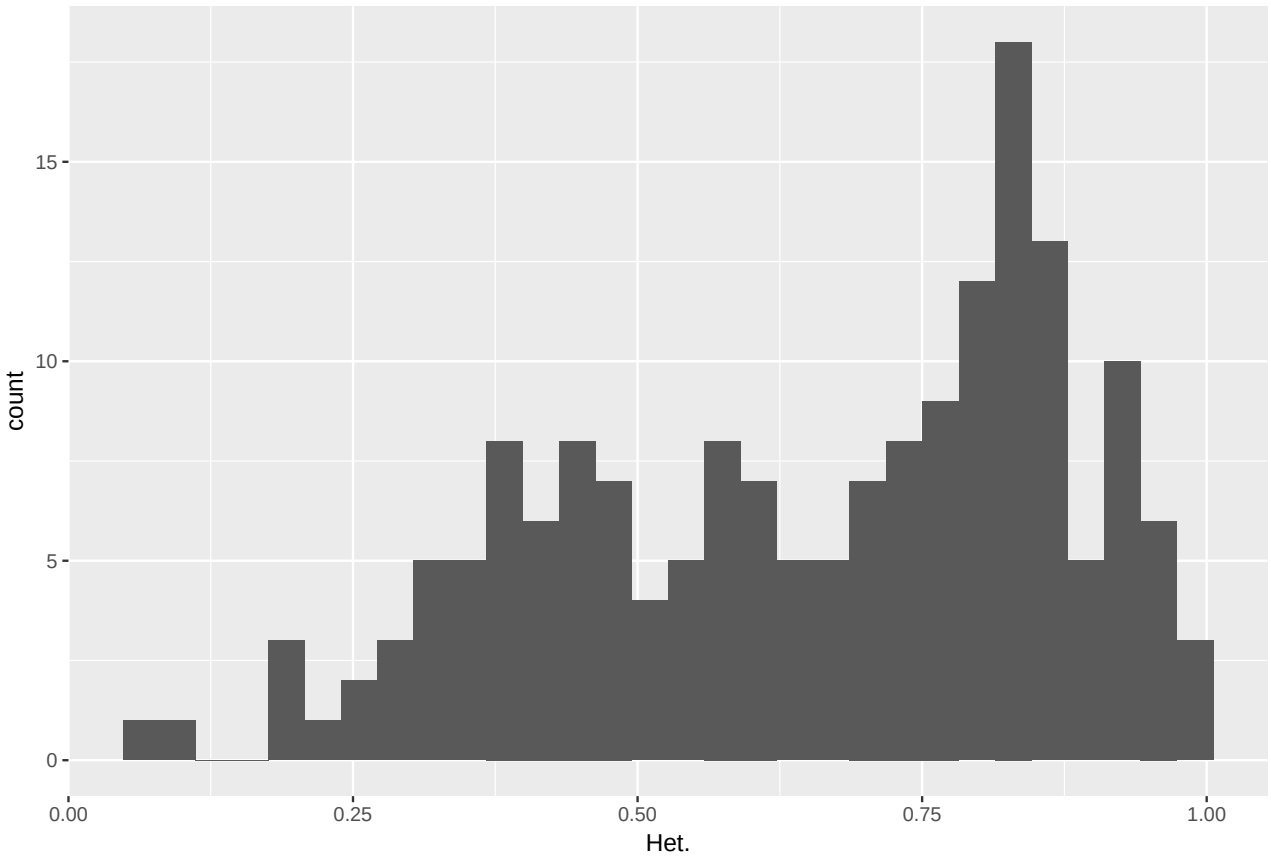

Figure 7.13: Distribution of heterozygosity in centromeric windows across all individuals from libraries 7, 7b and 10b. Centromeric windows contain all SNPs located within 25% of the maximum distance to centromere in each chromosome. All chromosomes were pooled together for each individuals using a weighted average where the weight of each chromosome is given by the number of SNPs contained in the window.

Chapter 8

Male-Female Fst

Date: 17.07.2017
During preliminary analyses, I noticed recurrent peaks in Fst between diploid males and females in different families. This could indicate the presence of male-deleterious alleles. To further investigate this possibility, I analysed Fst again using all individuals (including haploids).

I first looked at Fst over all 6 chromosomes in the assembly, averaging the Fst at each SNP across family to have an overview (Figure 8.1), and then splitting the families to identify recurrent peaks (Figure 8.2).

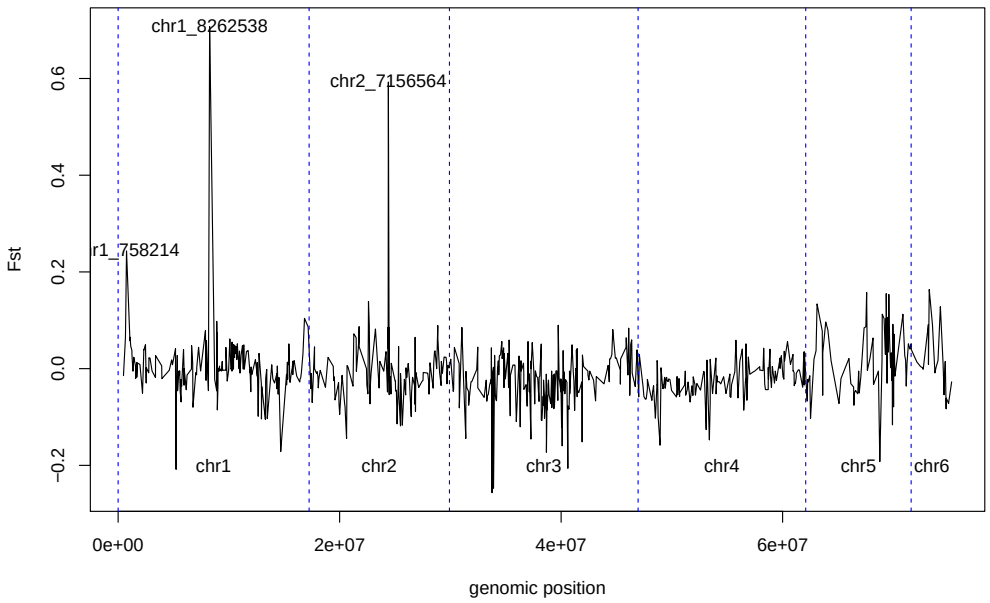

Figure 8.1: Fst values averaged at each SNP across all families. All individuals, including haploids and mothers are included in the analysis and top scoring peaks are labelled with the SNP position in the format "chromosome_basepair" where basepair is the position within the chromosome. Note: populations parameters set to : min depth (D)=20 and prop pop (r)=80

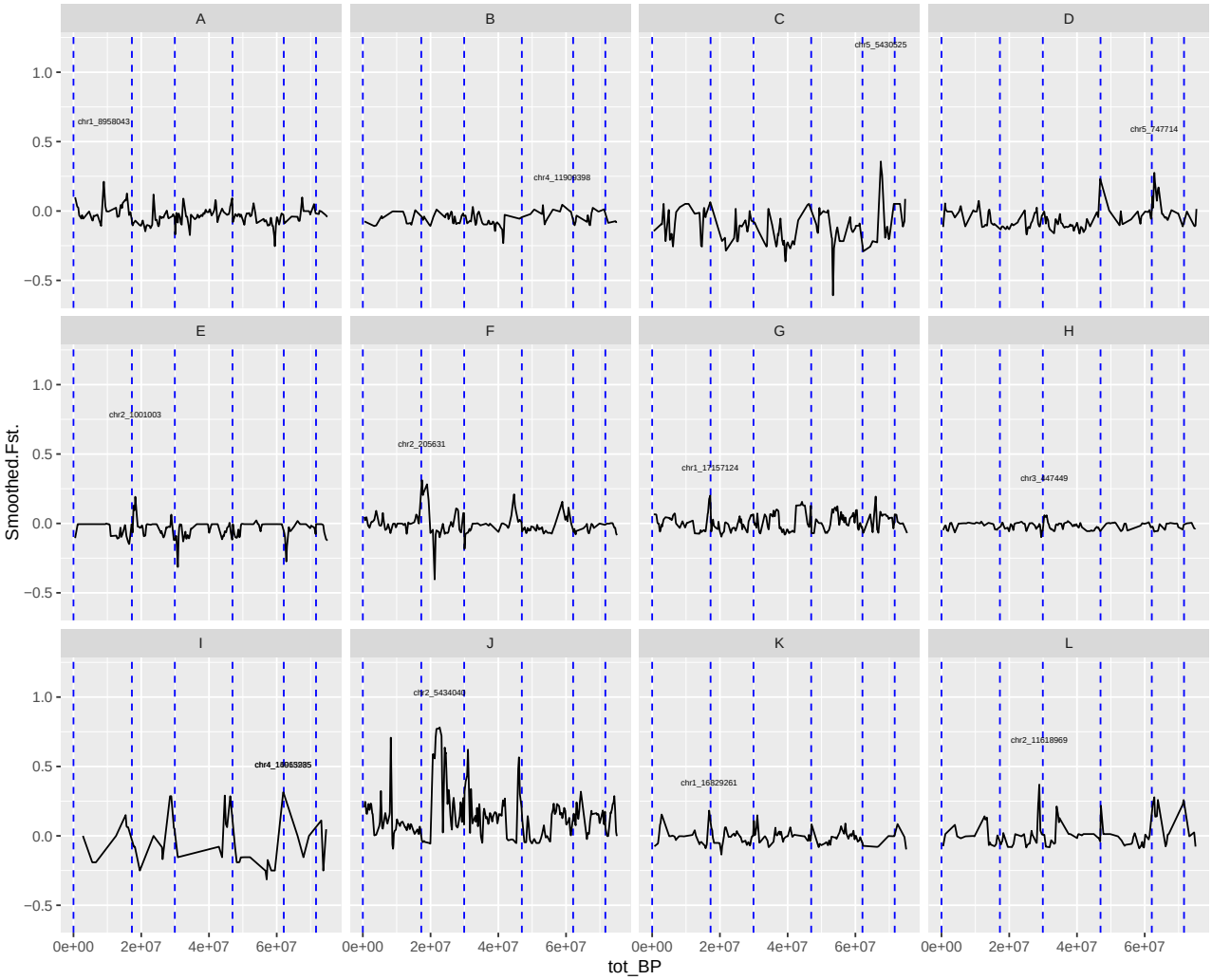

Figure 8.2: Fst values at each SNP by family. All individuals, including haploids and mothers are included in the analysis and top scoring peaks are labelled with the SNP position in the format "chromosome_basepair" where basepair is the position within the chromosome. Note: populations parameters set to : min depth (D)=20 and prop pop (r)=80

The issue with averaging the Fst over families as in figure 8.1 is that some SNPs can be sequenced in only one family, therefore the averaged value will only depend on that family. If we merge all families' curve into a single plot, it appears there is no single peak common to all families. This can either mean that there is no interesting signal, or if that the signal depends on the family's genetic background, possibly due to an epistatic effect. Here, however, I do not have the sample size to test for it.

This issue will be dropped from the analysis as the effect is much weaker than what we expected at first and probably just random noise.

Chapter 9

Association mapping

9.1 Coarse analyses

Before running any association mapping analysis, I will scan the genome for region that are frequently homozygous in males and heterozygous in females. This is done only after removing SNPs that are homozygous in mothers and individuals that are defined as haploids with fixed homozygosity threshold from the populations analysis.

Date: 21.08.2017

The statistic I use when scanning the genome for CSD is $CSD = \frac{Hom_m + Het_f}{2}$ where Hom_m and Het_f are the proportions of homozygous males and heterozygous females respectively at each SNP. If the value is 1, all individuals respect the CSD pattern (all males are homozygous and all females are heterozygous) if it is 0, no individuals respect it (all males are heterozygous and all females homozygous). This metric should not be biased by the gradual decrease in heterozygosity along the chromosome arms as the same weight is given to males and females regardless of numbers.

9.2 Case-control association test

Date: 18.08.2017 I will use a simple case-control odds ratio to identify CSD hits. I will first do it without taking families into account. I will then perform this test separately for each category of mother and finally I will use distance to centromeres to refine the lists of hits (c.f. Chapter: 7 Number of CSD loci").

The odds ratio are computed fo each SNP using the following observed (O) contingency table where each cell is the number of genotypes corresponding to the category:

	Homozygous	Heterozygous	Total
Male	Mo	Me	Mt
Female	Fo	Fe	Ft
Total	To	Te	Tt

Accordingly, the expected (E) numbers are given by:

	Homozygous	Heterozygous	Total
Male	$\frac{Mt*To)}{Tt}$	$\frac{Mt*Te)}{Tt}$	Mt
Female	$\frac{Ft*To)}{Tt}$	$\frac{Ft*Te)}{Tt}$	Ft
Total	To	Te	Tt

The χ^2 can then be measured as follows:

$\chi^2 = \frac{(O(Mo)-E(Mo))^2}{E(Mo)} + \frac{(O(Me)-E(Me))^2}{E(Me)} + \frac{(O(Fo)-E(Fo))^2}{E(Fo)} + \frac{(O(Fe)-E(Fe))^2}{E(Fe)}$

I applied this approach on grouped (i.e. single population run with all families) pooled (proportion of homozygous individuals calculated using all families for each SNP) data to have a first insight on the potentially interesting regions (Figure 9.1). If I want to get reliable results however, I will need to use a different association mapping technique that takes family information into account and to perform the test separately on each category of family (see section: 7.1, "Categorizing mothers").

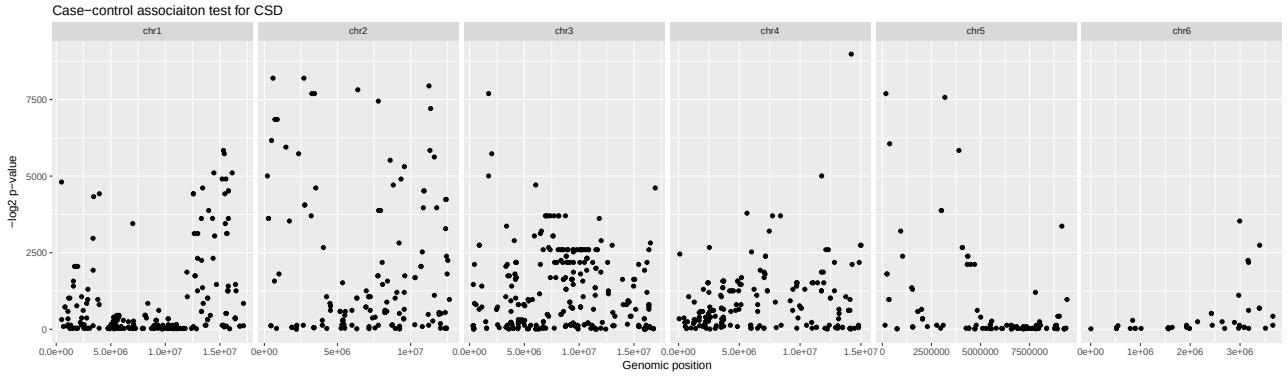

Figure 9.1: Manhattan plots with log2 p-values for the case-control association test on CSD. Populations was run on all families together and for each SNP, all individuals were pooled to compute the proportion of homozygous individuals

Date: 31.08.2017
The chi2 test is not appropriate in this case especially since I have small values in my cells, I will use Fisher-exact test instead.
Date: 01.09.2017
Using association mapping with the Fisher-exact test and pooling new indiviudals (including those from library 10b) together yields a few candidate regions and seems promising (Figure 9.2). I tried it both with the Benjamini-Hochberg correction (Figure 9.3) and the bonferroni correction (Figure 9.4). The next step will be to perform this association mapping per category and to identify the genes in the candidate regions.

9.2.1 Case-control on categories

Here, I apply the test after subsetting individuals by categories, inferred using diploid male production as detailed in section 7.1 "Categorizing mothers".

I will do it considering different number of categories, corresponding to different scenarios regarding the number of CSD loci. Figure 9.1 already addresses the unlikely scenario of single-locus CSD. Here, I will deal with 2 and 3 loci scenarios, with 3 and 7 categories, respectively.

9.2.2 Refining hits using centromeres

9.3 Family-based association mapping

I will try to adapt the "generalized family-based association test for dichotomous traits" (GDT) published by Chen W.-M., Manichaikul A. and Rich S.S. in The American Journal of Human Genetics (2009). This test should work both in inbred and non-inbred families and takes family information into account.

Here, I will modify the test presented in the publication to test for association of homozygosity with male phenotype instead of a particular allele with a disease. I will also need to make it work with a single parent.

Date 23.08.2017

After investigating the proofs of the score used in the family based test mentioned above, I realized adapting it would not make sense; it uses pedigree information and relies on the differences in kinship coefficients or

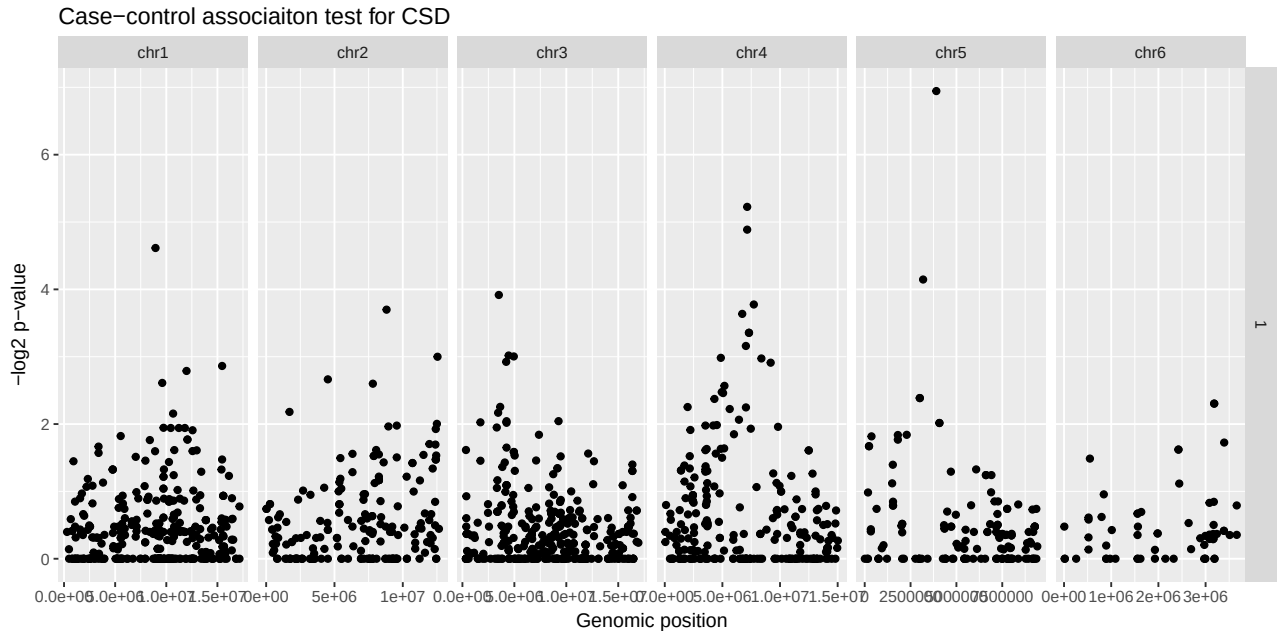

Figure 9.2: Manhattan plots with log10 p-values for the case-control association test on CSD using Fisher exact test. Populations was run on all families together and for each SNP, all individuals were pooled to compute the proportion of homozygous individuals. P-values are not corrected for multiple testing. P-value thresholds of 0.05 and 0.01 are displayed using horizontal black and red lines respectively.

proportion of IBD alleles. Since I have asexuals, all kinship coefficient in the families are 1, they are all siblings and it does not make much sense to include pedigree information. I will instead focus on a better implementation of the case-control test.

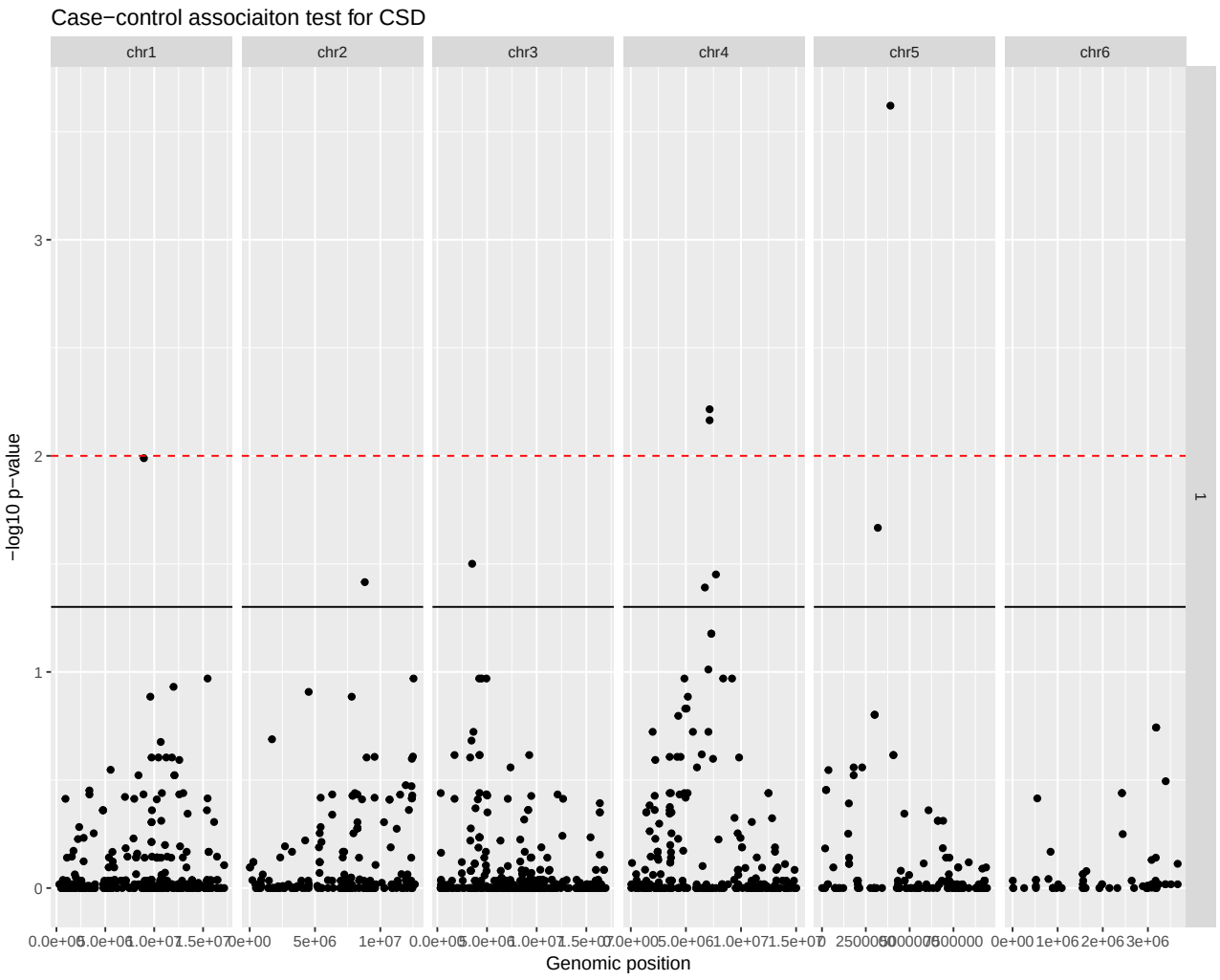

Figure 9.3: Manhattan plots with log10 p-values for the case-control association test on CSD using Fisher exact test. Populations was run on all families together and for each SNP, all individuals were pooled to compute the proportion of homozygous individuals. Benjamini-Hochberg correction for multiple testing was applied. P-value thresholds of 0.05 and 0.01 are displayed using horizontal black and red lines respectively.

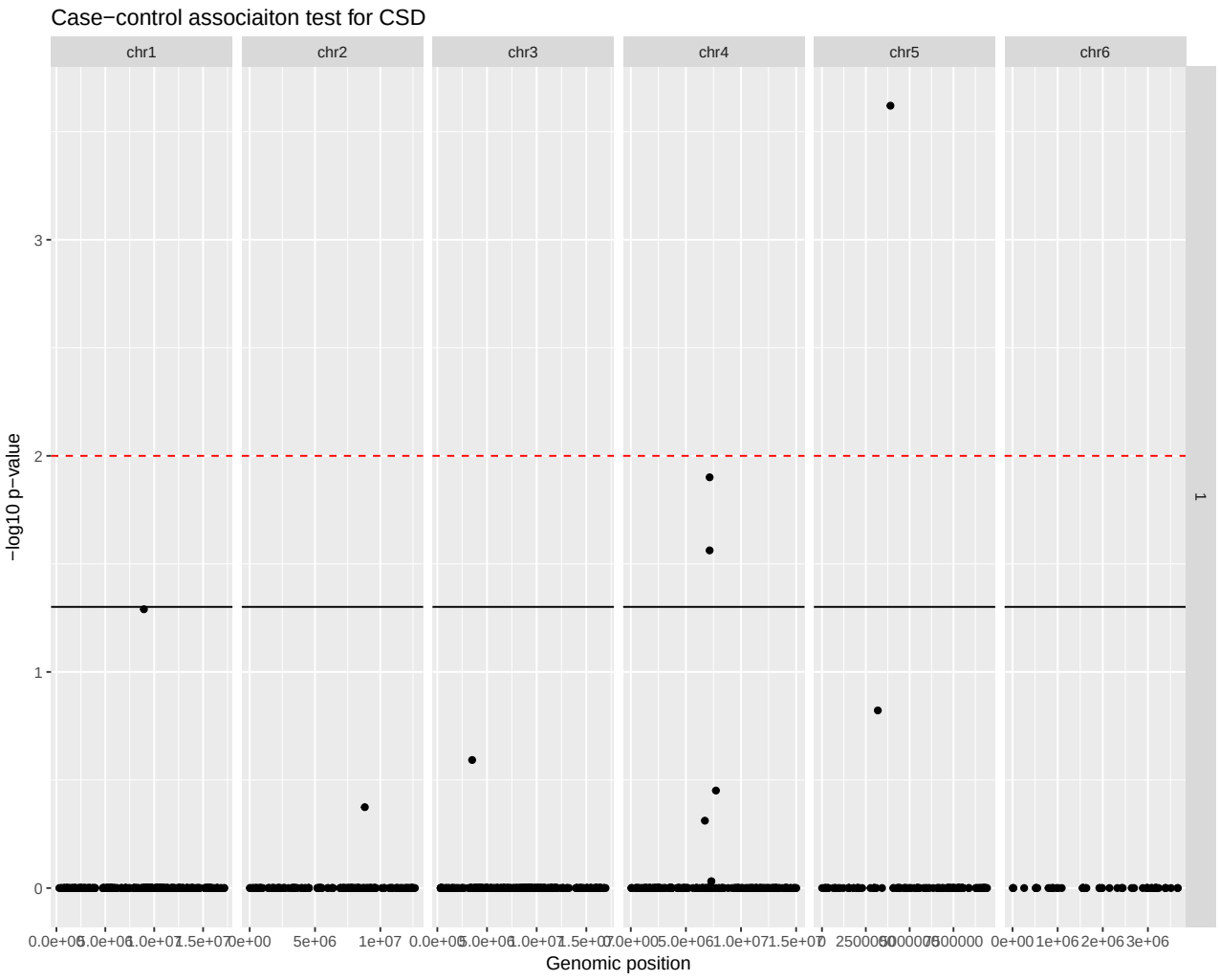

Figure 9.4: Manhattan plots with log10 p-values for the case-control association test on CSD using Fisher exact test. Populations was run on all families together and for each SNP, all individuals were pooled to compute the proportion of homozygous individuals. Bonferroni correction for multiple testing was applied. P-value thresholds of 0.05 and 0.01 are displayed using horizontal black and red lines respectively.

Chapter 10

Coverage analysis

In this section, I check wether there are individuals or genomic regions that have abnormally high or low coverage, I identify the potential reasons for these abnormalities and eventually, I will remove the concerned elements from the analysis.

10.1 Individual stats

Prior to coverage, I had a look at the raw reads statistics. On average,each sample has 803951 reads and the (ordered) reference genome is 140.7 Mb long. This makes for an average of $5.7 * 10^{-3}$ reads per basepair. In comparison, the 3 datasets used in the Paris 2017 paper (Roadmap to STACKS) have $9.6 * 10^{-4}$, $2.7 * 10^{-3}$ and $8.6 * 10^{-3}$.

I first analysed the coverage on a per individual basis. That means for each individual, it has been averaged over all polymorphic SNPs kept throughout the STACKS pipeline. The mean sample coverage is relatively high and follows a normal distribution centered around 119X with a standard deviation of 33. There are two outliers with exceptionally high coverage (at 228X and 270X, respectively) which I might consider removing. The coverage is not different across families (Figure **??**). Coverage is not affected by the family of an individual (it is not affected by ploidy or sex either).

Analyzing the number of sites (SNPs) per sample revealed that it was strongly affected by the family (Figure 10.2). This could be caused by storage duration, reagent quality or manipulations. If so, I cannot account for this and will either need to get rid of lowest quality families (such as I) or get more individuals.

Update: 06.08.2017

To investigate whether the inter-family variation in the number of sequenced sites was due to differing amounts of sequenced DNA, I compared the number of sequencing reads per family (Figure 10.3). There is interfamily variation, but it seems relatively low, compared to the number of sites in figure 10.2. This variation is likely due to the low number of samples and is probably not meaningful. Moreover, the order of families, when sorted from highest to lowest is different for number of sequenced sites and number of reads.

10.2 Genome coverage

To identify regions of high or low coverage, I extracted information about the mean coverage per family at each polymorphic site that passed the STACKS populations filters. I then binned SNPs by 1kb regions (Figure 10.4). Some SNPs have extreme coverage values (¿600X) and might be caused by repetitive regions or paralogs that have been merged as a single locus.

10.3 Lowering the filters

Date: 09.08.2017

The main issues I am facing in this chapter are 1) the abnormally high coverage across my samples and 2) the high variation in the number of sites between families.

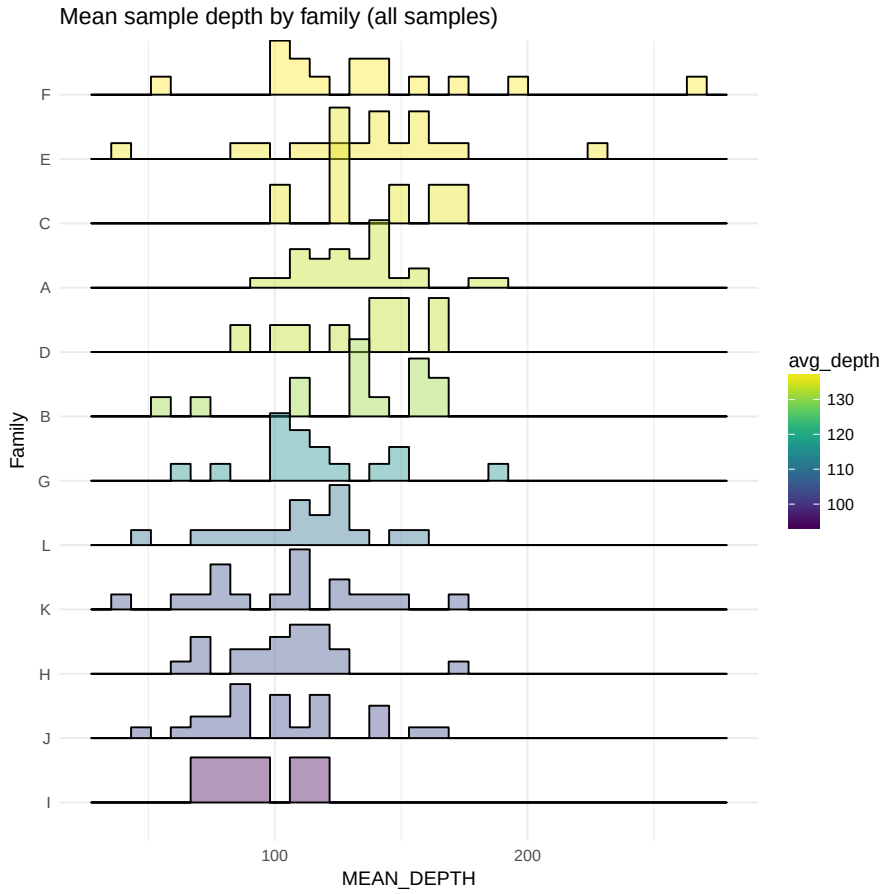

Figure 10.1: Sample depth, averaged across all polymorphic sites which passed STACKS populations filters. Individuals of each family are plotted together and histograms are colored according to the family's mean depth.

The reason the high mean coverage per sample was worrying is because I have about the same amount of reads per sample as the 3 datasets described in the roadmap to stacks paper (Paris et al, 2017), yet i have at least twice more coverage per sample. After re-reading the paper, I found out this was simply a consequence of my stringent filters at the populations stage, which they do not use (see citation below).

"Despite controlling for the mini- mum read depth of alleles at the ustacks phase of the pipe- line, many studies also incorporate a minimum stack depth required for individuals at a locus in the populations mod- ule of STACKS (e.g. Gaither et al. 2015; Ivy et al. 2016; Kjeldsen et al. 2016). Such a method is undesirable, as read depth has already been accounted for by the SNP model. Once the SNP model has made a determination, its evaluation should be trusted and using further non-statistically based limits on depth of coverage is ill advised and will result in the arbitrary dropping of loci."

I will follow their advice since my coverage is absurdly high, and I would benefit a sharp increase in the number of loci by removing this filter.

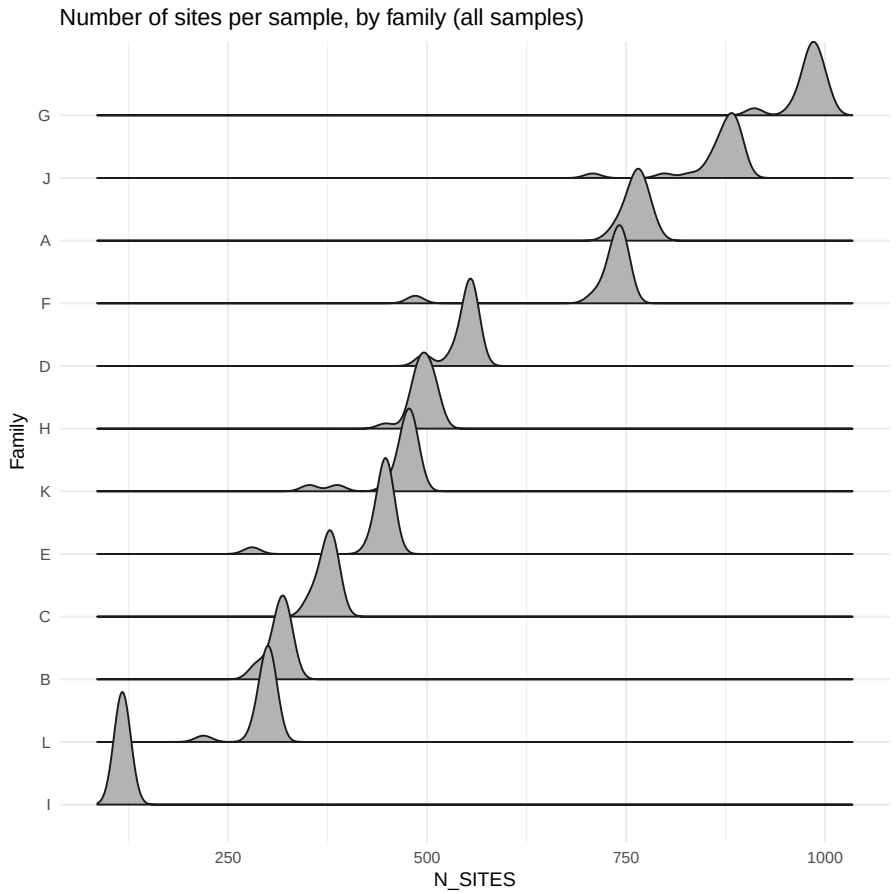

Figure 10.2: Number of polymorphic sites (SNPs) per individual that passed the STACKS populations filters. Distribution of individuals is shown by family. There is a strong clustering of families.

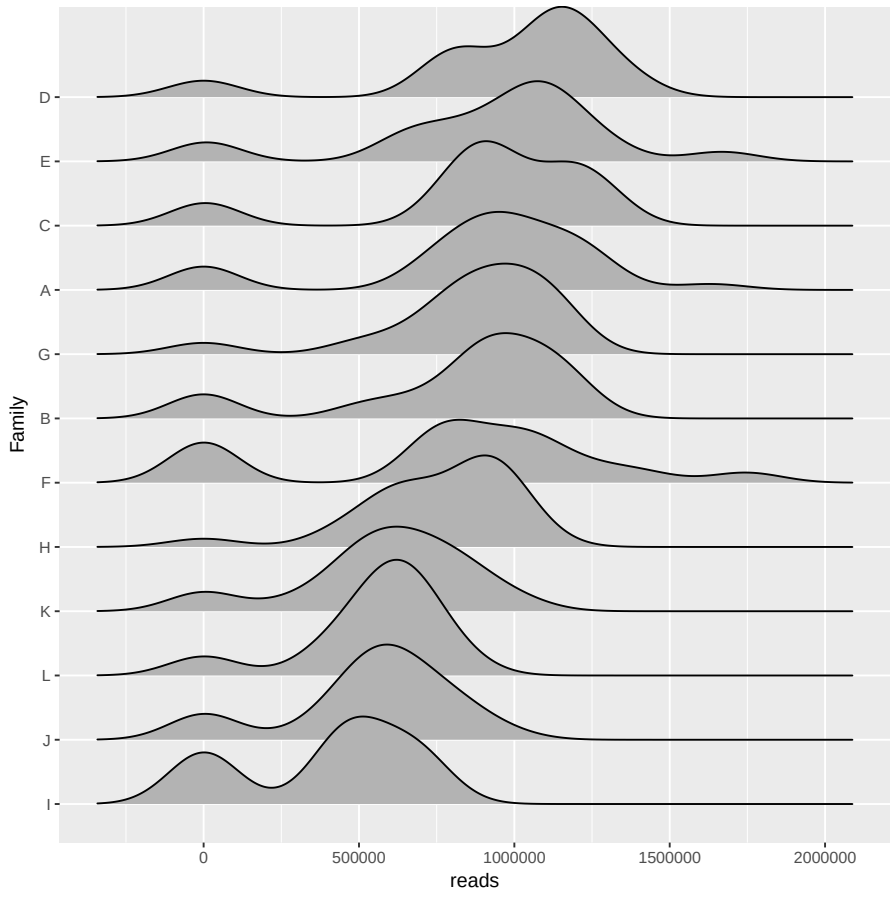

Figure 10.3: Number of sequencing reads per individual, before mapping or any filter. Distribution of individuals is shown by family. The inter-family variation is not particularly high relative to intra-family variation.

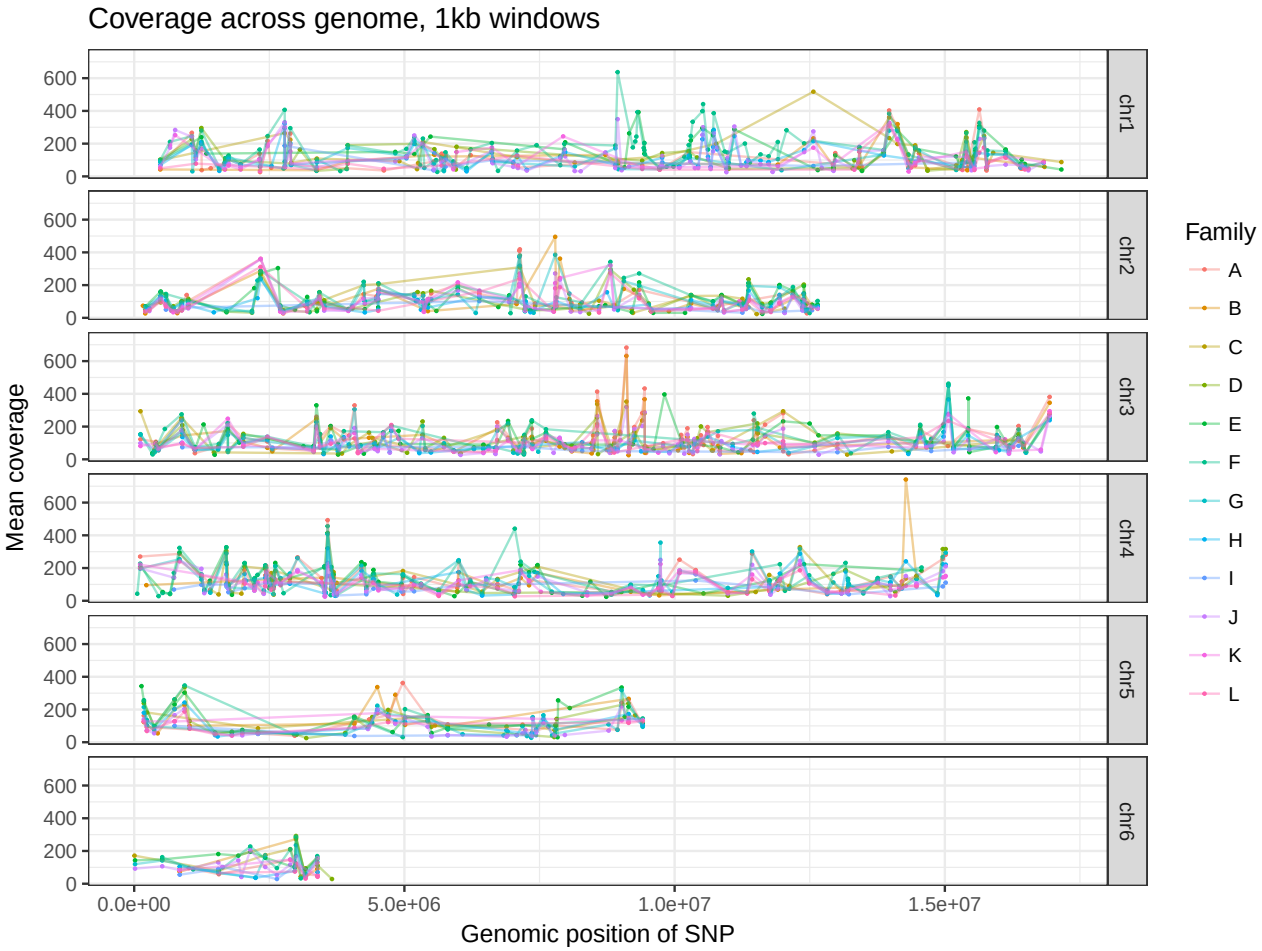

Figure 10.4: Mean depth per family at every SNP that passed the populations filters.

Chapter 11

Additional samples (lib10 + lib10b)

Date: 26.08.2017
Libraries 10 and 10b got back from the samples and I have processed them through the pipeline. Library10 suffered a dramatic loss in read count due to an unidentified technical problem (10x less reads than lib10b).
IMPORTANT: W10 appeared twice in the name list: once in library 10b and once in what will later be library11. Therefore I shifted the numbering by 1 for family W in library11. I will need to make sure this is the case in the barcode file as well.

Date: 28.08.2017
After removing all samples from library10 (see subsection 11.1.2 "Coverage", below). I still have NNN diploid individuals, distributed across FFF families.

11.1 Data description

11.1.1 Family composition

With the new samples, I now have 239 diploids across 32 families. Many families are relatively small, however (Figure 11.1) and relying on sex ratio among diploid offspring to categorize families could be unreliable.

11.1.2 Coverage

The coverage across genome (Figure 11.3) did not change much by adding the new libraries, however looking at the coverage per individuals(Figure 11.2) and number of site per individuals (Figure 11.4) confirms the poor quality of library 10. These individuals barely passed the filters i have put in place, but overall they still have very low read counts and including them exposes my whole analysis to technical biases. I will therefore remove all samples from that library. Note that I'll also remove W10 since it is the only healthy sample left in the family and all females are lost (they were in lib10).

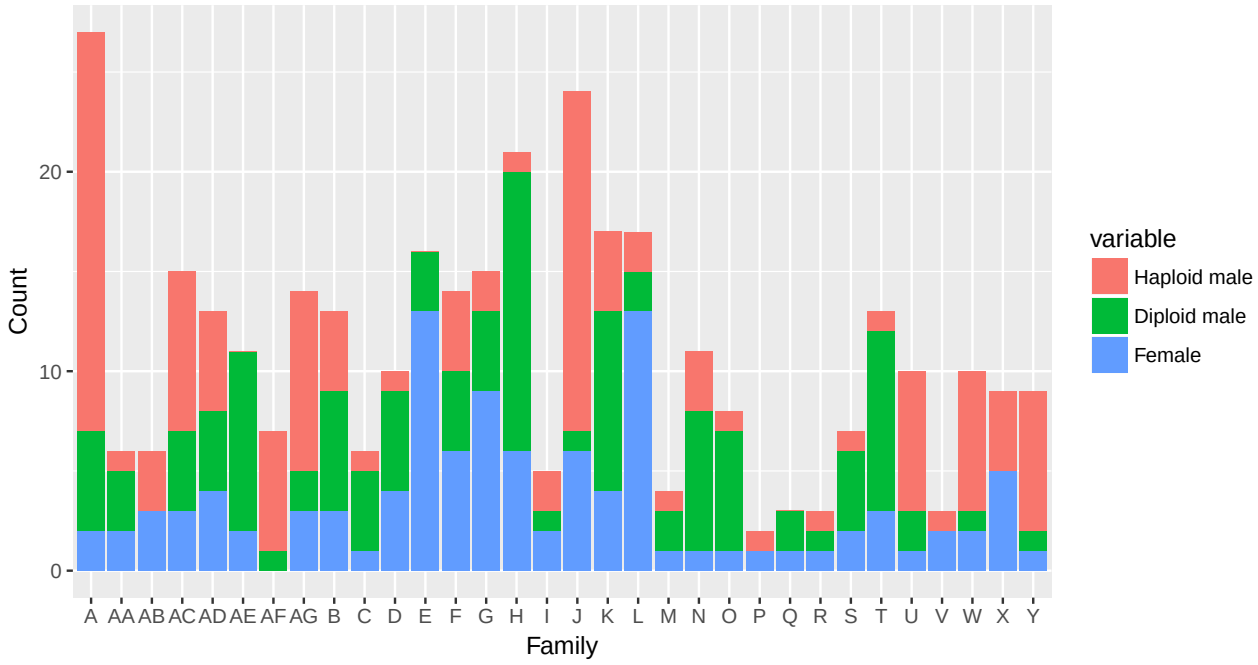

Figure 11.1: Composition of offspring per family when including new samples from library 10 and 10b.

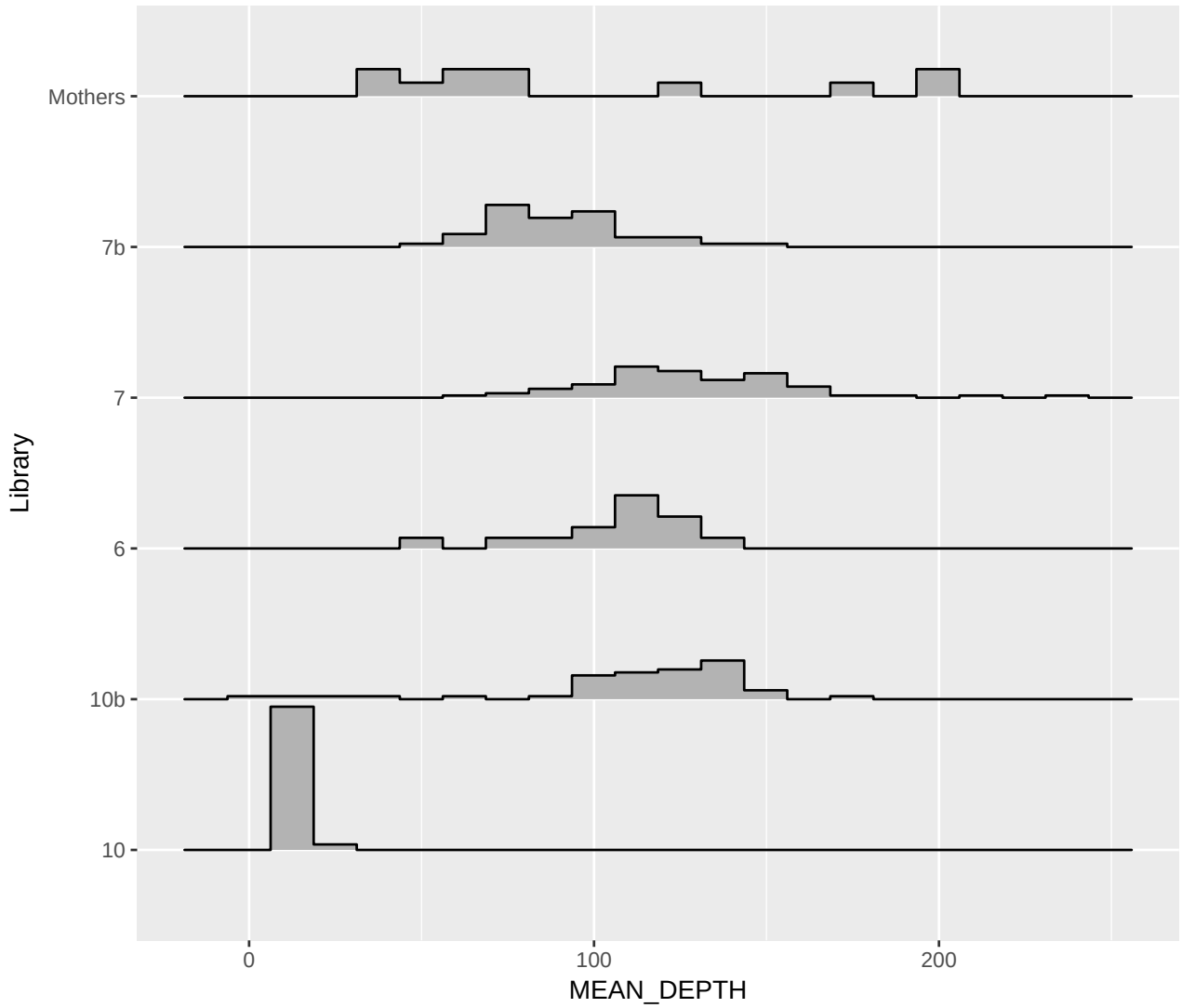

Figure 11.2: Distribution of the mean sequencing depth per individuals in each library including the new ones (10 and 10b).

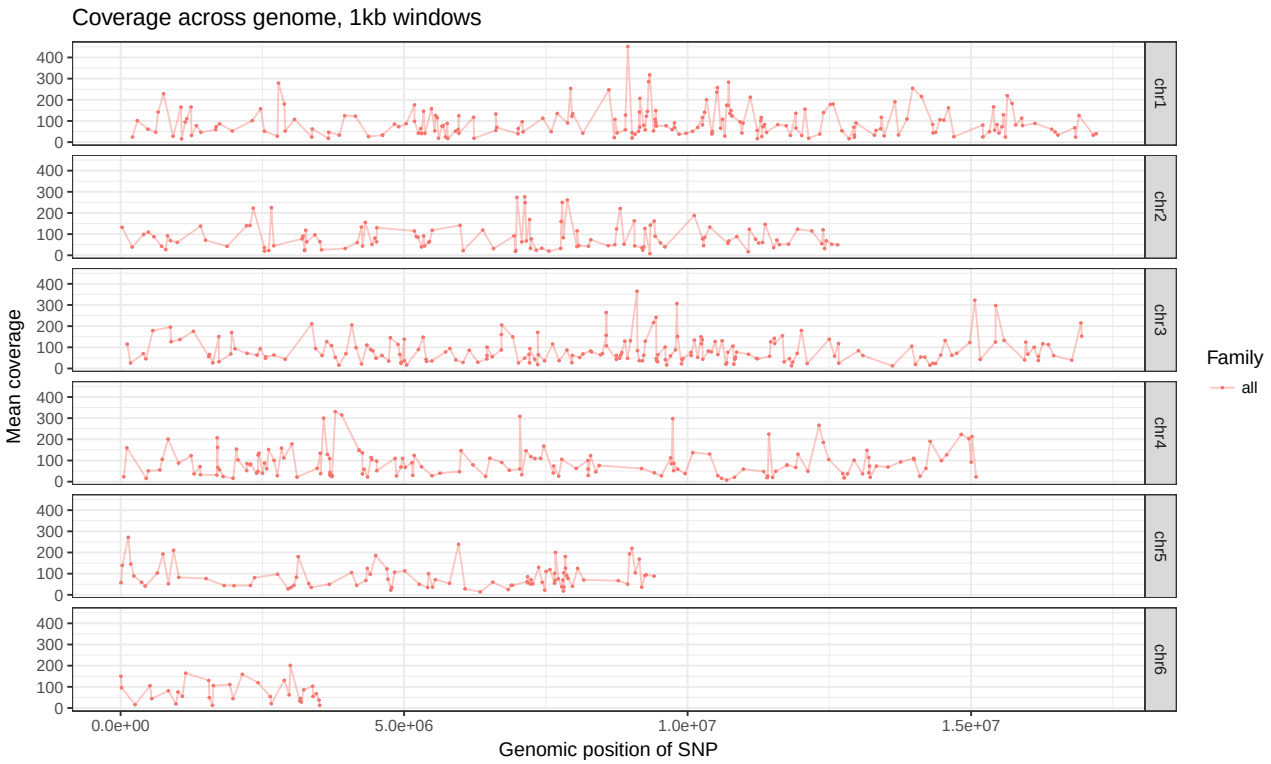

Figure 11.3: Sequencing depth across genome. Depth is averaged over all samples at each sites and values are averaged by 1kb windows. New samples from libraries 10 and 10b are included

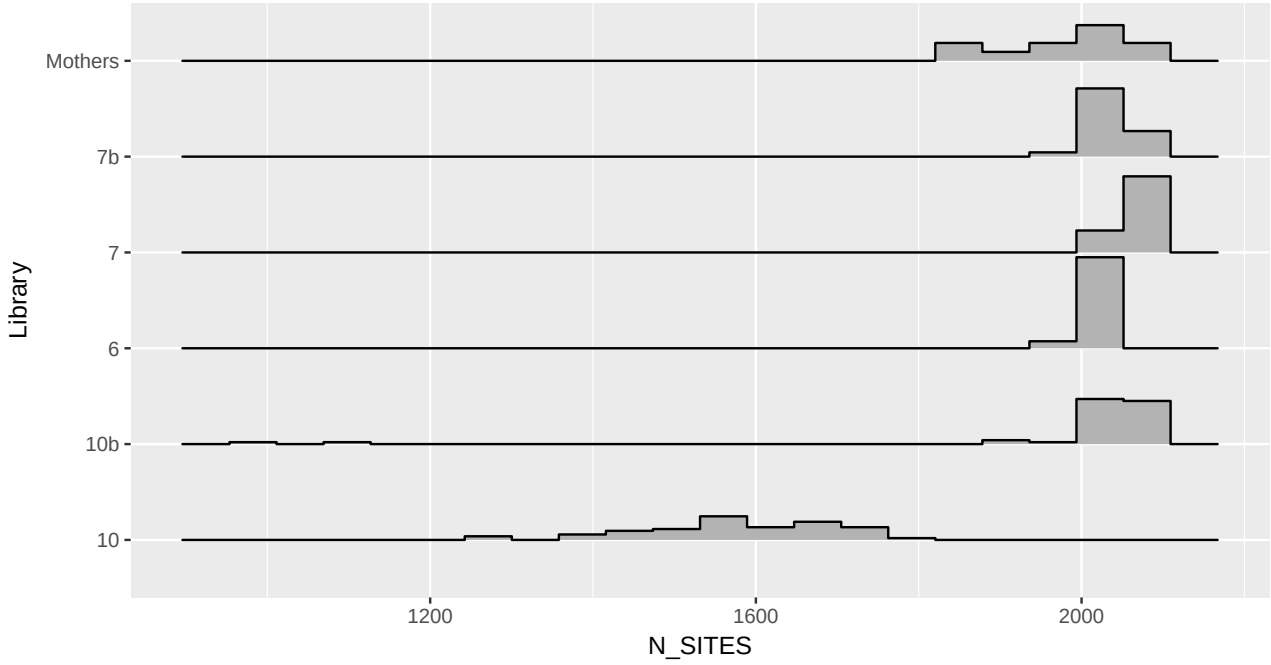

Figure 11.4: Distribution of the number of sites per individuals in each library including the new ones (10 and 10b).

Chapter 12

Pipeline reorganization

Date: 11.10.2017

In order to increase the number of loci and the quality of my data, I will follow the advice from the roadmap to stacks paper and use a Ustacks + later mapping approach rather than Mapping + pstacks. For convenience, I would like to keep both options in the final pipeline, and I would also add the possibility to run it without an LSF cluster. To allow these changes, I needed to reorganize the flow of my pipeline.

12.1 Reorganization

Originally, I used a Makefile with all steps of the pipeline. I want to be able to swap all STACKS-related steps without rewriting the entire Makefile or duplicating it. The approach I used is to outsource all STACKs steps into a separate Makefile and call that secondary makefile using a rule in the main Makefile. This allows to have as many different STACKS strategies which each have their own Makefile.

I currently use Makeref for the Pstacks pipeline and Makenovo for the Ustacks pipeline (Figure 12.1).

12.2 New STACKs strategy

It is shown in the roadmap to stacks paper that building stacks de novo and mapping them to the genome later on yields better results. I will therefore adopt that approach and use GSNAP to map reads, although I might try different softwares later on.

The new *de-novo* approach will work as follows:

- Build stacks *de-novo* with Ustacks
- Build catalog with Cstacks
- Match samples to catalog with Sstacks
- Run samples through populations
- Map consensus sequences that passed populations filters with GSNAP
- integrate alignment information into catalog

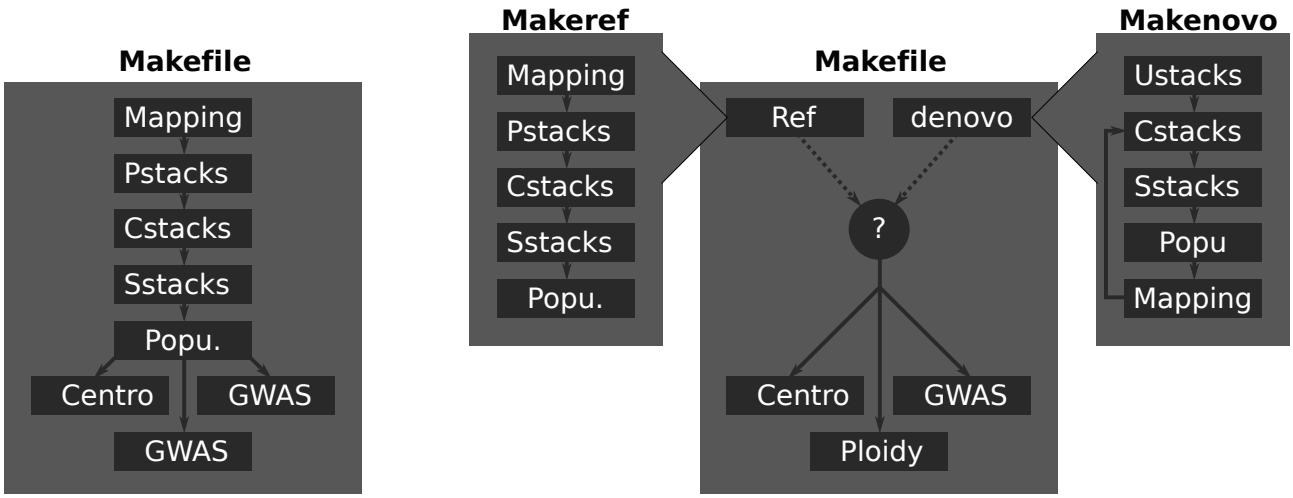

Figure 12.1: Original structure of the pipeline (left) with data flow and dependencies, versus new structure (right) where STACKS-related tasks are delegated to separate Makefiles

Chapter 13

Final samples

Date: 16.10.2017

After receiving the second run of sequencing data for libraries 12 and 12b, I pooled them for demultiplexing and ran the usual STACKS pipeline with stringent parameter for splitting by ploidy (min locus depth 20). The one thing I changed in the pipeline is that I removed the max_obs_het filter which excluded loci heterozygous in more than 90% of individuals. I will replace this filter by a blacklist of loci heterozygous in more than 50% of haploid males after inference of ploidy. The distribution of homozygosity per individuals revealed a highly homozygous cluster of individuals from the D cross forming a second mode close to the haploids (Figure 13.1). This is problematic because it will be hard to disentangle it from the haploid mode. After discussing with Casper, we concluded that it would be more appropriate to consider the D cross separately as it could also cause the catalog to exclude important loci in the C cross (due to the R=0.8 filter) from the catalog.

Running the populations program will all individuals together instead of per-family removed the second mode and revealed that most individuals from the D-cross are haploids (Figure 13.2). This highlights the potential risks of running populations per family. Because this cross seems to have a quite divergent genetic background, it will be excluded from the analysis and analyzed separately.

13.1 Pipeline changes

Populations will now only be run with all individuals pooled to avoid non-overlapping sets of SNPs and discrepencies in homozygosity levels across families. Individuals from cross C and D will be run completely separately and I will probably not used at all given the high number of haploids and low number of females in that cross. I also stopped using the max_obs_het filter in populations, in favor of a personalized method to blacklist loci found to be heterozygous in at least given proportion of haploids.

13.2 Splitting crosses

I separated individuals from cross D from those in C or R and ran them separately through the whole pipeline.

13.2.1 Cross D

This cross contains 186 individuals, of which 174 made it through the pipeline. They are split into 15 families.

- Mean number of sites per sample: 864
- Average locus depth per sample: 108
- Ploidy count: 35 diploids, 139 haploids

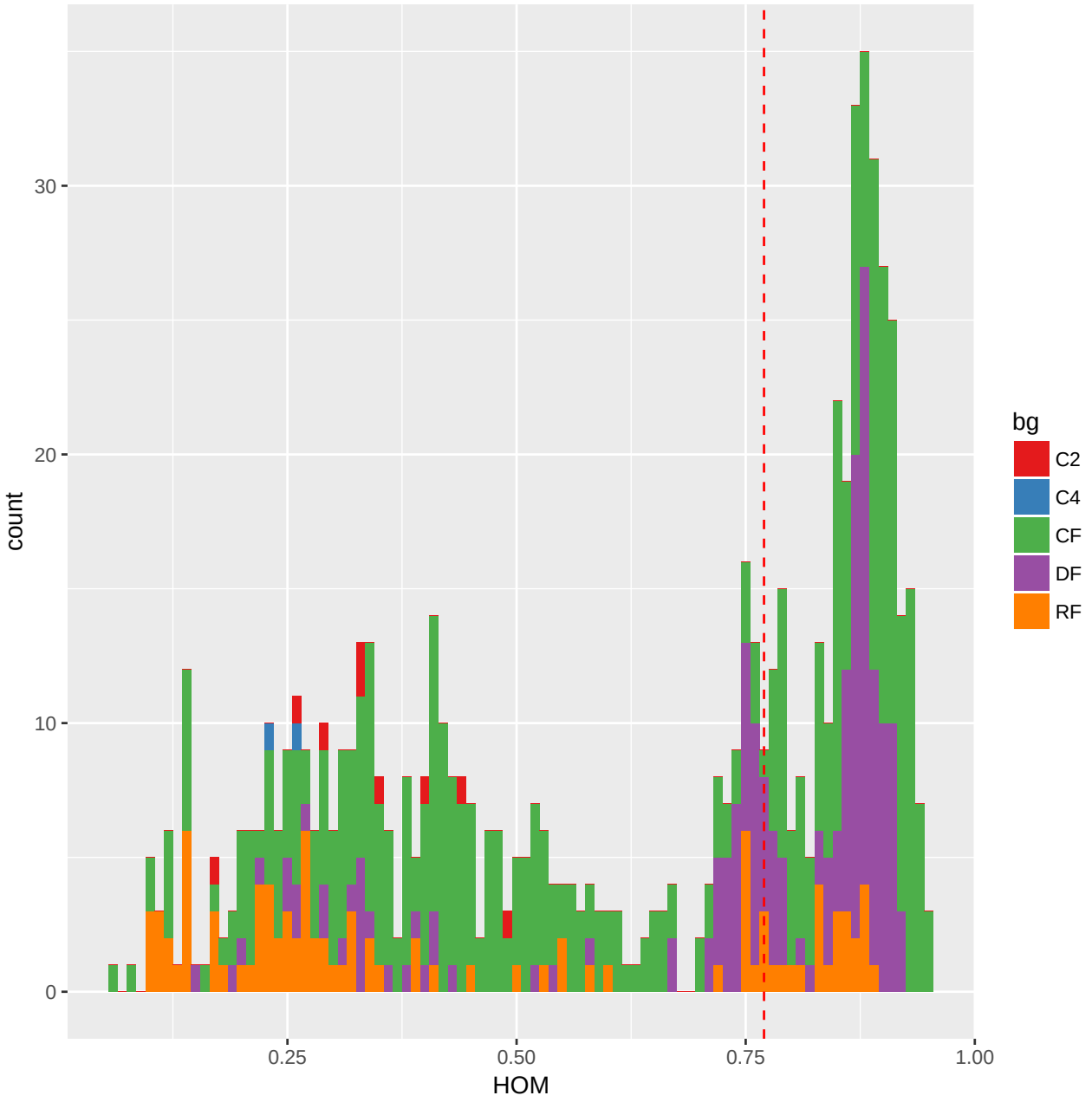

Figure 13.1: Distribution of the proportion of homozygosity among variant sites per individuals. Populations was run per-family without max-obs-het filter. Individuals are colored based on background.

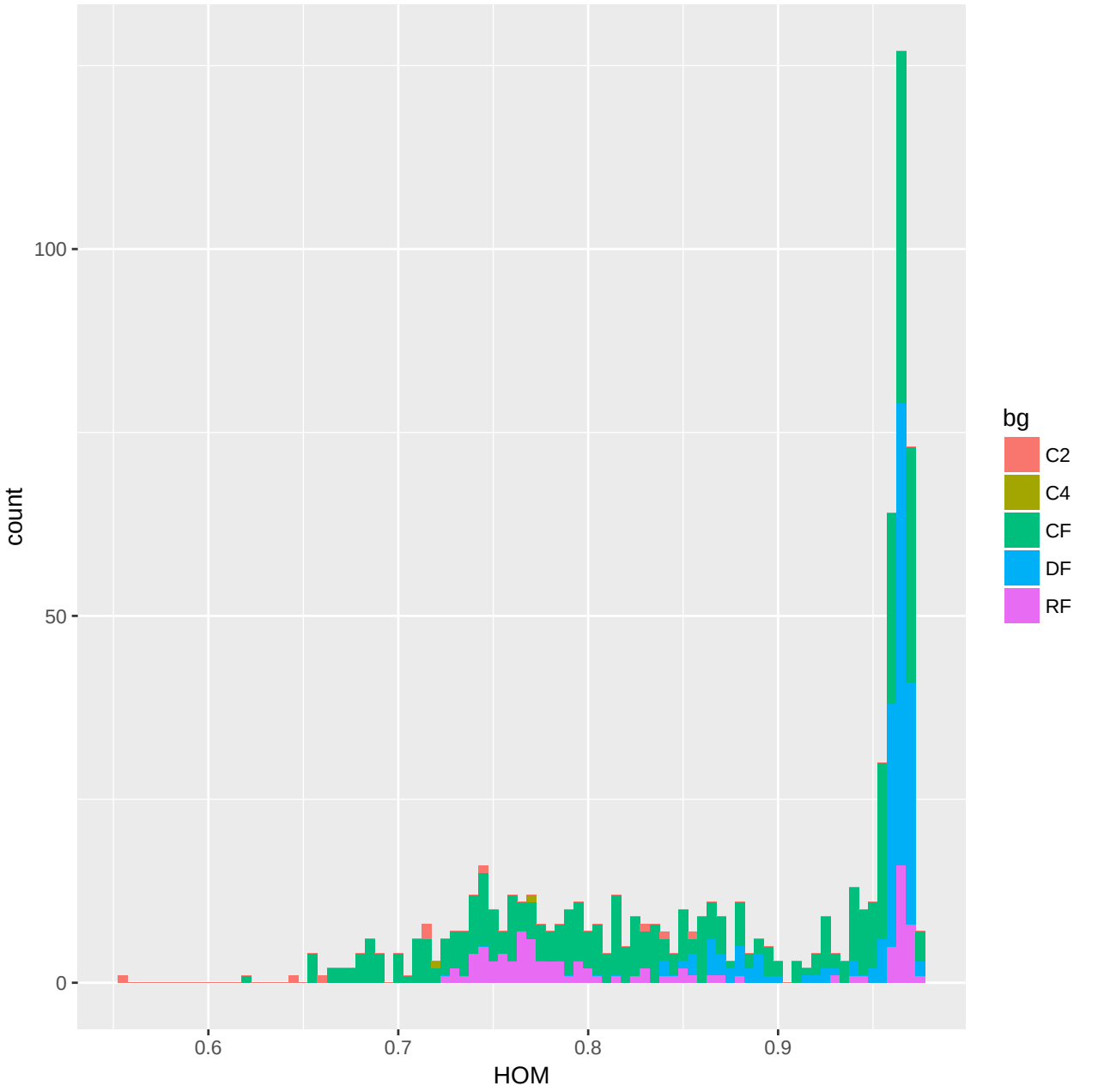

Figure 13.2: Distribution of the proportion of homozygosity among variant sites per individuals. Populations was run with all individuals together without max-obs-het filter. Individuals are colored based on background.

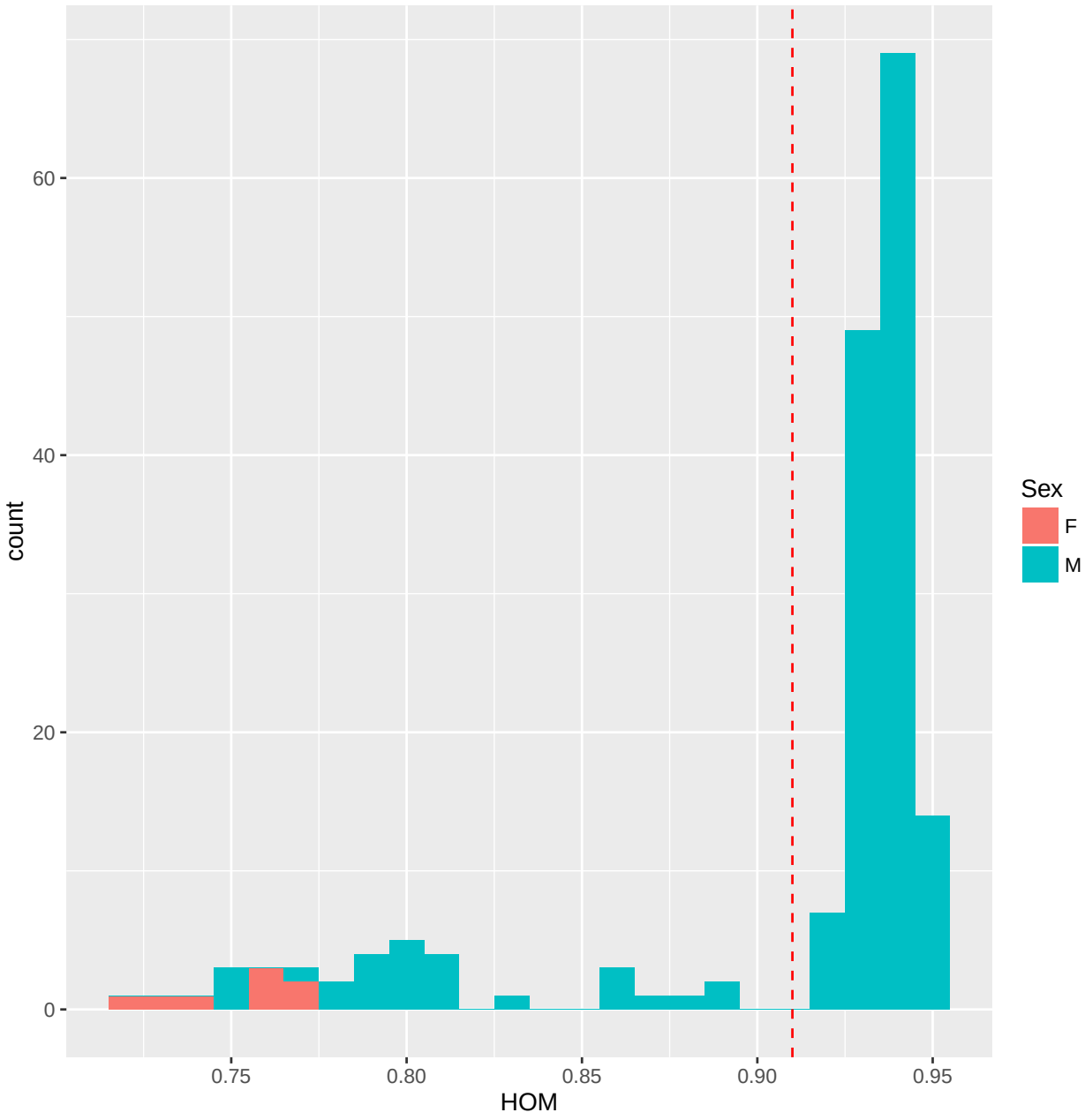

Figure 13.3: Distribution of the proportion of homozygosity among variant sites per individuals. Populations was run with all individuals from cross D together without max-obs-het filter. Individuals are colored based on sex. The red dashed vertical line represent the arbitrary (conservative) threshold I defined earlier to separate ploidy from grouped populations runs.

13.2.2 Cross C

This cross contains 588 individuals, of which 530 made it through the pipeline. They are split into 45 families.

- Mean number of sites per sample: 1439
- Average locus depth per sample: 127.
- Ploidy counts: 314 diploids, 216 haploids

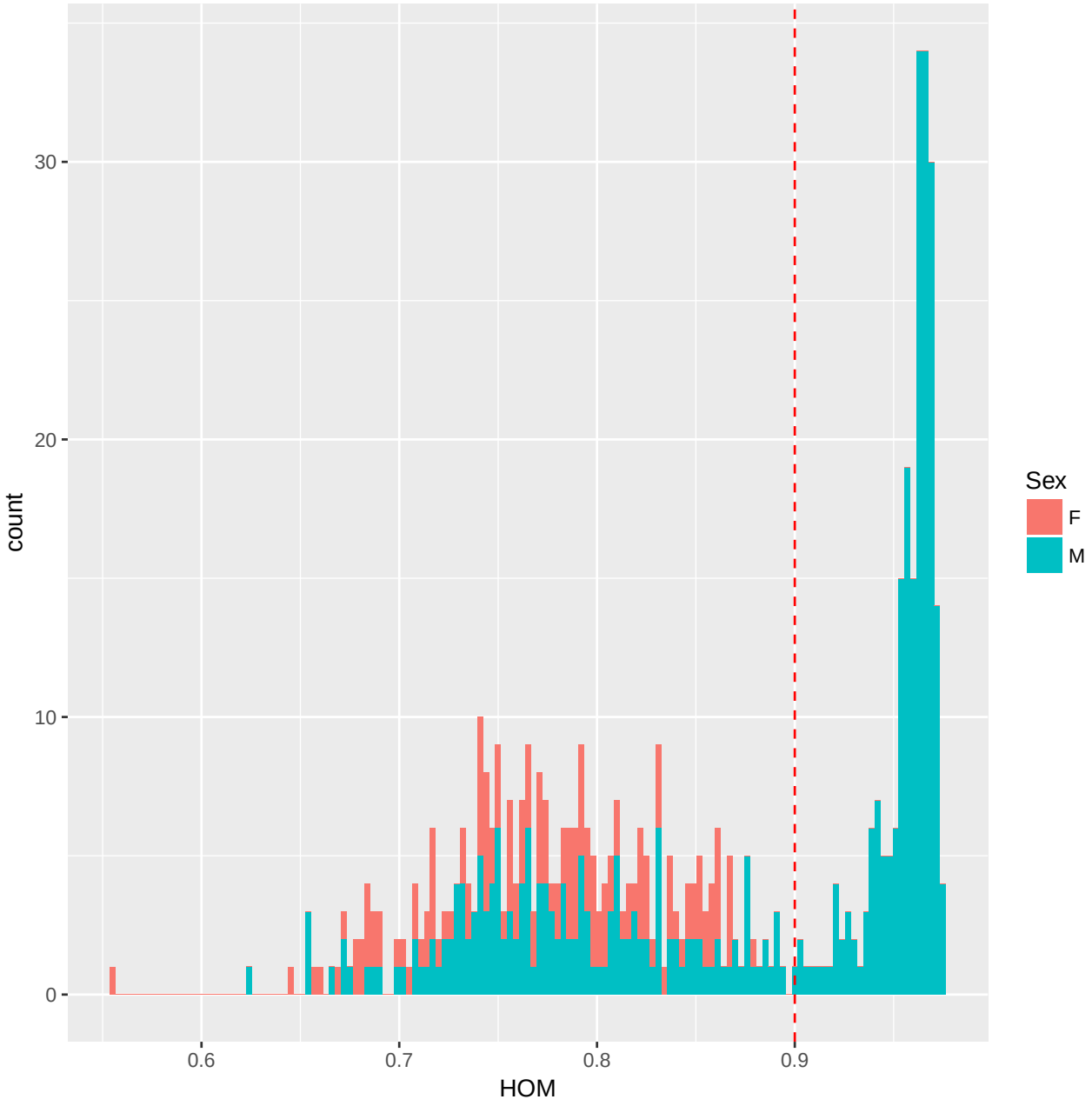

Figure 13.4: Distribution of the proportion of homozygosity among variant sites per individuals. Populations was run with all individuals from cross C together without max-obs-het filter. Individuals are colored based on sex. The red dashed vertical line represent the arbitrary (conservative) threshold I defined earlier to separate ploidy from grouped populations runs.

13.3 CSD

13.3.1 Finding centromeres

The centromere positions remained quite similar to previous estimations when including the last samples (Figure 13.5).

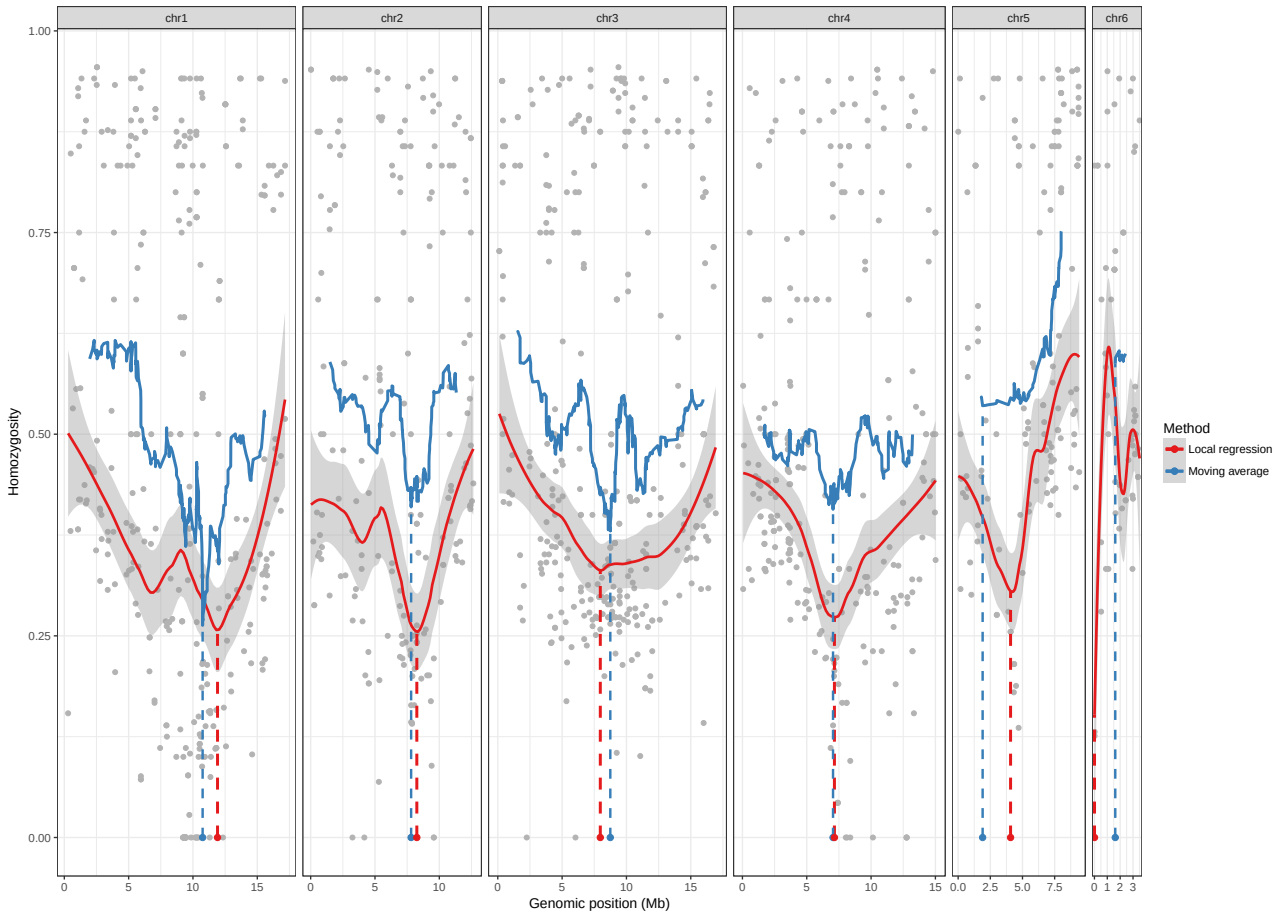

Figure 13.5: Centromere mapping using both a simple moving average (window size 50 SNPs) and a local regression of degree 2 with span 0.6 and weights according to individuals per SNP.

13.3.2 Association mapping

We still observe 3 peaks in the association mapping when including the last samples (Figure 13.6).

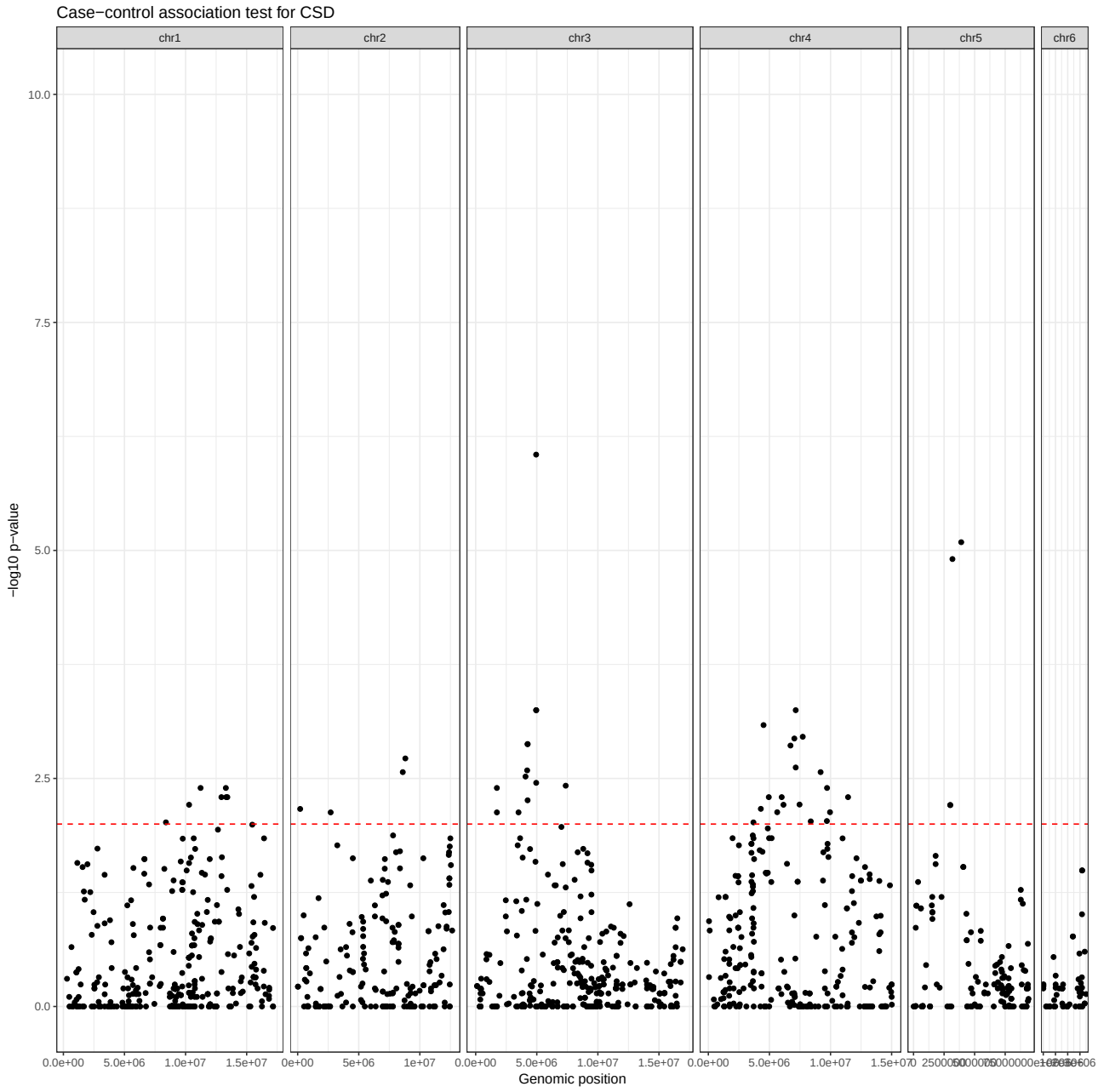

Figure 13.6: Case-control association mapping for CSD using association mapping. One-sided Fisher exact test was performed for each SNP with the alternative hypothesis that proportion of heterozygous individuals is higher among females than males. P-values are corrected using FDR with the Benjamini-Hochberg procedure.

Chapter 14

Annotations and homologies

The last step of the analysis is to determine what genetic elements might be responsible for the CSD mechanism. I will use annotations generated by the BIPAA on their *L. fabarum* genome. They inferred elements using maker and retrieved generated homology-based annotations with blast2go. Because the assembly I am using was reordered using a linkage map, I had to convert the coordinates of their annotations ot our ordered assembly (i.e. into chromosomes instead of contigs).

14.1 Annotations

I found no transformer homolog in any of the chromosomes containing CSD hits, however transformer-2 is present in chromosome 2, where there is no association mapping hit. There was no protein annotation common to all 3 candidate region when looking 200kb around the significant hits of GWAS(Figure 14.1), but there are many uncharacterized proteins (see table below for all annotations within 200kb of significant hits). The next step would be to look at transcriptome annotations from RNA-seq data in larvae, in case there are unannotated/noncoding transcripts.

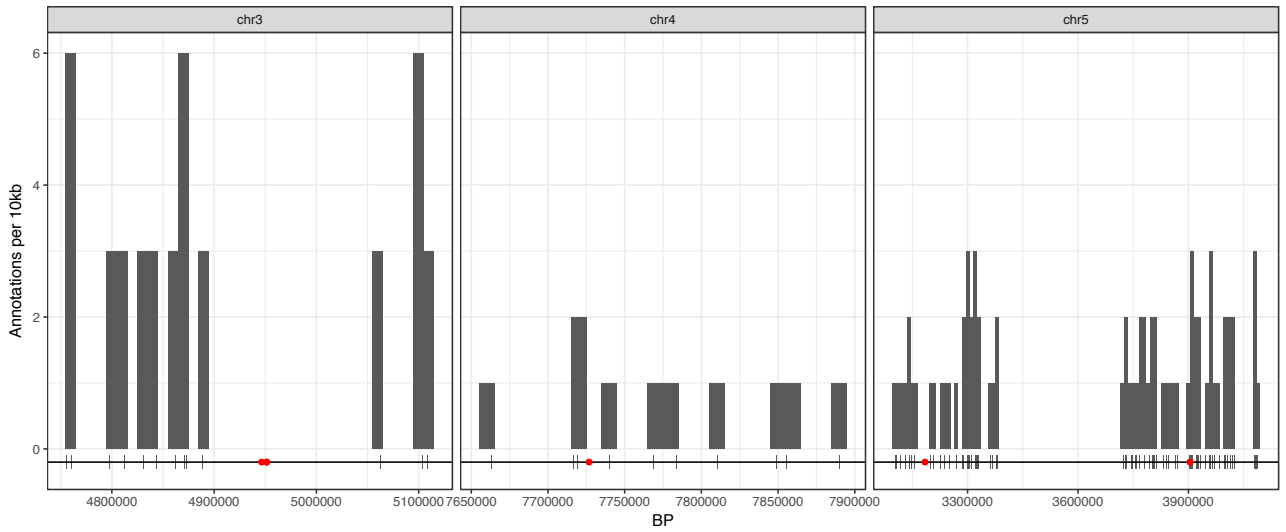

Figure 14.1: Visualization of annotation density in the neighbourhood (200kb) of significant GWAS hits. Red dots are the significant hits and grey vertical lines are the annotated proteins. Histograms show the number of annotations in 10kb bins.

chrom	start	end	ID	GO	term
chr3	4755773	4756151	LF003046-PA	GO:0005975	chitinase-like protein idgf4

chrom	start	end	ID	GO	term
chr3	4760373	4760594	LF003045-PA	GO:0016021	golgi integral membrane protein 4-like isoform x1
chr3	4796906	4797511	LF003043-PA	GO:0016021	uncharacterized protein loc100879956
chr3	4811951	4812047	LF003042-PA	GO:0016021	adenylate cyclase type 6
chr3	4831089	4831238	LF003041-PA	GO:0016021	adenylate cyclase type 6
chr3	4843615	4843781	LF003040-PA	GO:0016020	proton-coupled folate transporter
chr3	4861774	4861814	LF003038-PA	GO:0016020	vasoactive intestinal polypeptide receptor 1-like isoform x1
chr3	4870994	4871049	LF003037-PA	GO:0005507	protein sco1 mitochondrial
chr3	4872792	4872935	LF003036-PA	GO:0032266	vacuolar protein-sorting-associated protein 36
chr3	4888350	4888940	LF003033-PA	GO:0016021	uncharacterized protein loc107272897
chr3	5062396	5062706	LF003027-PA	GO:0006457	prefoldin subunit 5
chr3	5103157	5103384	LF003024-PA	GO:0016020	hig1 domain family member mitochondrial
chr3	5103863	5104279	LF003023-PA	GO:0046872	protein rufy3
chr3	5108354	5108613	LF003022-PA	GO:0046872	run and fyve domain-containing protein 2 isoform x3
chr3	4755773	4756151	LF003046-PA	GO:0005975	chitinase-like protein idgf4
chr3	4760373	4760594	LF003045-PA	GO:0016021	golgi integral membrane protein 4-like isoform x1
chr3	4796906	4797511	LF003043-PA	GO:0016021	uncharacterized protein loc100879956
chr3	4811951	4812047	LF003042-PA	GO:0016021	adenylate cyclase type 6
chr3	4831089	4831238	LF003041-PA	GO:0016021	adenylate cyclase type 6
chr3	4843615	4843781	LF003040-PA	GO:0016020	proton-coupled folate transporter
chr3	4861774	4861814	LF003038-PA	GO:0016020	vasoactive intestinal polypeptide receptor 1-like isoform x1
chr3	4870994	4871049	LF003037-PA	GO:0005507	protein sco1 mitochondrial
chr3	4872792	4872935	LF003036-PA	GO:0032266	vacuolar protein-sorting-associated protein 36
chr3	4888350	4888940	LF003033-PA	GO:0016021	uncharacterized protein loc107272897
chr3	5062396	5062706	LF003027-PA	GO:0006457	prefoldin subunit 5
chr3	5103157	5103384	LF003024-PA	GO:0016020	hig1 domain family member mitochondrial
chr3	5103863	5104279	LF003023-PA	GO:0046872	protein rufy3
chr3	5108354	5108613	LF003022-PA	GO:0046872	run and fyve domain-containing protein 2 isoform x3
chr3	4755773	4756151	LF003046-PA	GO:0005975	chitinase-like protein idgf4
chr3	4760373	4760594	LF003045-PA	GO:0016021	golgi integral membrane protein 4-like isoform x1
chr3	4796906	4797511	LF003043-PA	GO:0016021	uncharacterized protein loc100879956
chr3	4811951	4812047	LF003042-PA	GO:0016021	adenylate cyclase type 6
chr3	4831089	4831238	LF003041-PA	GO:0016021	adenylate cyclase type 6
chr3	4843615	4843781	LF003040-PA	GO:0016020	proton-coupled folate transporter
chr3	4861774	4861814	LF003038-PA	GO:0016020	vasoactive intestinal polypeptide receptor 1-like isoform x1
chr3	4870994	4871049	LF003037-PA	GO:0005507	protein sco1 mitochondrial
chr3	4872792	4872935	LF003036-PA	GO:0032266	vacuolar protein-sorting-associated protein 36
chr3	4888350	4888940	LF003033-PA	GO:0016021	uncharacterized protein loc107272897
chr3	5062396	5062706	LF003027-PA	GO:0006457	prefoldin subunit 5
chr3	5103157	5103384	LF003024-PA	GO:0016020	hig1 domain family member mitochondrial
chr3	5103863	5104279	LF003023-PA	GO:0046872	protein rufy3
chr3	5108354	5108613	LF003022-PA	GO:0046872	run and fyve domain-containing protein 2 isoform x3
chr4	7662977	7662991	LF013183-PA	GO:0003677	homeobox protein hmx
chr4	7716629	7717162	LF014636-PA	GO:0006259	uncharacterized protein loc105842614
chr4	7719367	7719680	LF014637-PA	GO:0006259	uncharacterized protein dwil'gk22850
chr4	7739899	7740140	LF010545-PA	GO:0005737	uncharacterized protein loc107265729
chr4	7768779	7769106	LF010548-PA	GO:0004725	tyrosine-protein phosphatase 69d
chr4	7783681	7784055	LF010550-PA	GO:0016020	required for meiotic nuclear division protein 1 homolog
chr4	7810568	7810970	LF010552-PA	GO:0003676	zinc finger protein 429-like isoform x3
chr4	7848658	7849135	LF010555-PA	GO:0016779	dna polymerase zeta catalytic subunit
chr4	7855355	7855459	LF010556-PA	GO:0005834	guanine nucleotide-binding protein subunit gamma-1
chr4	7889903	7890097	LF012290-PA	GO:0000166	hspb1-associated protein 1
chr5	3104517	3104912	LF009381-PA	GO:0008236	serine protease
chr5	3108040	3108175	LF009383-PA	GO:0016798	myrosinase 1-like
chr5	3116758	3117130	LF009386-PA	GO:0016021	uncharacterized protein loc411888 isoform x1
chr5	3131420	3131610	LF009387-PA	GO:0006351	aryl hydrocarbon receptor nuclear translocator-like protein 1 isoform x1
chr5	3141323	3141587	LF009389-PA	GO:0050999	nitric oxide synthase-interacting protein homolog
chr5	3143060	3143572	LF009390-PA	GO:0004930	protein srek1ip1-like
chr5	3146140	3146867	LF009391-PA	GO:0016021	neuropeptide y receptor-like
chr5	3155654	3155898	LF009394-PA	GO:0006457	peptidyl-prolyl cis-trans isomerase
chr5	3199652	3199830	LF007432-PA	GO:0006470	tyrosine-protein phosphatase non-receptor type 2 isoform x1
chr5	3206915	3207414	LF007434-PA	GO:0003924	hbs1-like protein
chr5	3226585	3227226	LF007439-PA	GO:0003676	hemk methyltransferase family member 2
chr5	3236823	3237162	LF007442-PA	GO:0016021	uncharacterized protein loc103572787
chr5	3249984	3250160	LF007444-PA	GO:0005840	60s ribosomal protein l36
chr5	3271009	3271058	LF007448-PA	GO:0016787	myrosinase 1
chr5	3286072	3286241	LF007451-PA	GO:0006004	alpha-l-fucosidase isoform x1
chr5	3288412	3288456	LF007452-PA	GO:0016021	cytosolic non-specific dipeptidase
chr5	3300652	3301513	LF007456-PA	GO:0003779	beta-parvin
chr5	3302820	3303027	LF007457-PA	GO:0005840	28s ribosomal protein mitochondrial
chr5	3304171	3304212	LF007458-PA	GO:0016020	uncharacterized protein loc106793526
chr5	3305988	3306711	LF007459-PA	GO:0044699	udp-sugar transporter ust74c-like
chr5	3310172	3310462	LF007460-PA	GO:0004713	discoidin domain-containing receptor 2-like
chr5	3320121	3321871	LF006413-PA	GO:0008026	atp-dependent dna helicase q1-like
chr5	3322345	3322573	LF006414-PA	GO:0071704	hydroxyacylglutathione mitochondrial isoform x1
chr5	3324976	3325014	LF006416-PA	GO:0008063	sorting nexin-16
chr5	3326542	3326757	LF006417-PA	GO:0016021	b(+)-type amino acid transporter 1-like
chr5	3329502	3329566	LF006418-PA	GO:0016021	b(+)-type amino acid transporter 1-like
chr5	3360783	3361337	LF006421-PA	GO:0006820	serine threonine-protein kinase mos
chr5	3366326	3368508	LF006424-PA	GO:0090501	endoribonuclease dcr-1
chr5	3378031	3378206	LF006425-PA	GO:0008270	e3 sumo-protein ligase pias2 isoform x2
chr5	3381501	3381630	LF006426-PA	GO:0016021	protocadherin-like wing polarity protein stan
chr5	3724567	3724820	LF004308-PA	GO:0003824	acyl- synthetase family member mitochondrial

chrom	start	end	ID	GO	term
chr5	3728983	3729346	LF004310-PA	GO:0007283	probable splicing factor 3b subunit 5
chr5	3732937	3733176	LF004311-PA	GO:0005488	bub3-interacting and glebs motif-containing protein znf207
chr5	3744015	3744382	LF004314-PA	GO:0007067	dynactin subunit 2
chr5	3747157	3747549	LF004315-PA	GO:0008440	inositol hexakisphosphate kinase 1 isoform x1
chr5	3755560	3758318	LF004316-PA	GO:0003779	huntingtin-interacting protein 1 isoform x2
chr5	3766086	3766334	LF004318-PA	GO:0046872	4-hydroxyphenylpyruvate dioxygenase
chr5	3768311	3768424	LF004319-PA	GO:0005622	probable nuclear transport factor 2 isoform x1
chr5	3779161	3779953	LF005975-PA	GO:0046872	uncharacterized protein loc107037375
chr5	3780850	3781113	LF005976-PA	GO:0005840	40s ribosomal protein s25
chr5	3793064	3793372	LF005979-PA	GO:0016773	sedoheptulokinase-like isoform x2
chr5	3802415	3802860	LF005983-PA	GO:0003676	zinc finger and scan domain-containing protein 12
chr5	3804498	3804968	LF005984-PA	GO:0008270	protein deltex
chr5	3807448	3808611	LF005985-PA	GO:0016787	ubiquitin carboxyl-terminal hydrolase 30 homolog
chr5	3814282	3814521	LF005989-PA	GO:0005975	beta-glucuronidase isoform x2
chr5	3832200	3832493	LF005992-PA	GO:0003824	phosphopantothenoylcysteine decarboxylase
chr5	3841319	3841587	LF005994-PA	GO:0031683	guanine nucleotide-binding protein subunit alpha homolog
chr5	3846210	3846422	LF005995-PA	GO:0030036	formin-like protein cg32138 isoform x2
chr5	3864664	3864675	LF005996-PA	GO:0030036	formin-like protein cg32138 isoform x1
chr5	3870961	3871267	LF005998-PA	GO:0016020	uncharacterized protein loc105433985 isoform x1
chr5	3902992	3903180	LF006000-PA	GO:0006260	replication factor c subunit 2
chr5	3905735	3906256	LF006002-PA	GO:0046872	superoxide dismutase
chr5	3907282	3907402	LF006003-PA	GO:0016491	superoxide dismutase
chr5	3912370	3912436	LF006004-PA	GO:0016020	cd63 antigen-like
chr5	3922179	3922291	LF006006-PA	GO:0003723	cbp80 20-dependent translation initiation factor isoform x1
chr5	3924113	3924505	LF006007-PA	GO:0016021	mitochondrial fission 1 protein
chr5	3927645	3927723	LF006008-PA	GO:0020037	dual oxidase isoform x1
chr5	3932611	3932932	LF006009-PA	GO:0016021	dual oxidase maturation factor 1
chr5	3945026	3945279	LF006013-PA	GO:0016021	peroxisomal membrane protein 2
chr5	3956512	3956708	LF006015-PA	GO:0090305	exosome complex component rrp40
chr5	3960325	3960525	LF006016-PA	GO:0016021	contactin-2 isoform x1
chr5	3964545	3965735	LF006018-PA	GO:0008757	probable 28s rrna (cytosine -c)-methyltransferase
chr5	3970714	3970826	LF006599-PA	GO:0016021	aquaporin-11
chr5	3984345	3984792	LF006603-PA	GO:0008233	sentrin-specific protease partial
chr5	3998103	3998234	LF006606-PA	GO:0005840	60s acidic ribosomal protein p2
chr5	4000033	4000216	LF006607-PA	GO:0035556	1-phosphatidylinositol -bisphosphate phosphodiesterase-like
chr5	4005625	4005710	LF006608-PA	GO:0016887	atp-binding cassette sub-family c member sur
chr5	4014860	4014985	LF006609-PA	GO:0030553	cgmp-dependent protein isozyme 1
chr5	4020932	4021132	LF006611-PA	GO:0008889	glycerophosphodiester phosphodiesterase domain-containing protein 1-like
chr5	4023955	4024395	LF006613-PA	GO:0005634	protein mago nashi homolog
chr5	4079398	4079748	LF006622-PA	GO:0016021	translocator protein
chr5	4081307	4081496	LF006623-PA	GO:0006468	serine threonine-protein kinase unc-51
chr5	4084649	4085149	LF006624-PA	GO:0006413	density-regulated protein homolog
chr5	4086667	4086867	LF006625-PA	GO:0005737	clustered mitochondria protein homolog

14.2 Homologies

I will look for sequence homology between candidate regions. There are several options to do so:

- Perform multiple alignment between candidate regions. Main drawback: arbitrary size of regions
- Blast whole genome against itself and find significant hits between regions. Main drawback: only pairwise comparisons, computationally inefficient
- Pairwise blast/alignment between chromosomes 3,4 and 5.

14.2.1 pairwise BLAST

Update: 04.11.2017 First strategy: Blasting chromosome 3 vs chromosome 4, 3vs5 and 4vs5. Using the following commands:

```
$ makeblastdb -in chr5.fasta -dbtype nucl
$ blastn -query chr4.fasta -db chr5.fasta -outfmt 6 -max_target_seqs 1 -out blast45.out
```

It seems multiple alignment programs such as CLUSTAL Omega cannot handle very long sequences (longer than 100kb). Therefore, the best strategy might be to blast regions against one another and then use an aligner on subsetted high scoring regions.

Detailed strategy (Discuss with Tanja and Casper):
1. Perform one-way pairwise blast between chromosomes. Example: 3v4, 4v5 and 3v5
1. Assign each alignment in blast output a unique ID: chrQ-chrS-ID. Example: chr3-chr4-13
2. Transform blast output into set of BED files, only including alignments with an E-value below 1e-10. Each

blast output generates 2 bed files: one with the query chromosome coordinates, the other with the subject chromosome coordinates.
Intervals in both files have the same IDs. Example: Blast output of chr3vchr4 produces $3v4.bed$ and $4v3.bed$
3. Perform pairwise intersections between bed files with the same coordinates system. Example: $3v4 \cap 3v5; 4v5 \cap 4v3; 5v3 \cap 5v4$
4. Each intersection operation yields 2 sets of IDs

14.2.2 MCScanX

Update: 11.11.2017

In stead, I used MCScanX to retrieve blocks of collinearity in the genome. This software only takes into account alignment between genes. The software takes a blast output and a bed file with all genes coordinates as input files. I proceeded as follows:

- Retrieve features coordinates from the BIPAA. These are gene predictions from MAKER in GFF format.
- Extract genes from the GFF file.
- Convert genes coordinates to our ordered assembly using custom python script.
- extract genes from ordered contigs only (i.e. placed in chromosomes)
- retrieve gene sequences with the getfasta utility from the BEDtools suite
- use BLASTN on the resulting fasta with outfmt 8 to get tabular output of all vs all genes blast.
- run MCScanX with the extracted genes coordinates in BED format and the BLASTN output.

  The resulting plots show several blocks of collinearity between chromosomes, but it does not entirely correspond with the CSD regions, especially for chromosome 5. This might be caused by the homology-based predictions of MAKER. Since the software relies on accurate gene prediction, I should use more reliable coordinates. I will therefore use the BIPAA RNAseq data to call genes instead of homology based predictions.

14.3 RNA-seq

To obtain more accurate gene coordinates for the collinearity analysis above, I will use transcripts coordinates obtained with RNA-seq. Besides, this will potentially uncover trancribed genes in candidate regions.

I obtained bam alignment files from RNA-seq reads from the BIPAA team. These reads were aligned to their unordered assembly with STAR.

I plan to assemble the transcriptome from the alignment file using cufflinks to have transcripts coordinates with isoforms.

Here are the issues I encountered so far:

- Being able to differentiate males and females would be useful to identify sex-specific expression levels and/or alternative splicing. Because these are larva, we don't know the sex, however it might be possible to identify haploid sample from heterozygosity levels using SNPs in transcriptome (?), or to cluster samples based on dominant isoforms in candidate CSD regions (if strong sample-specific isoform expression, interesting gene). Therefore, I want to know if it is possible to **classify reads in BAM files based on their original sample.**
- I don't have any info on the RNA-seq protocol: single-end or paired-end ? unstranded or strand specific. Looking at the BAM flags, I found no 1, therefore I assume this is single end sequencing. I also found about 33% of alignments had 16 (mapped to reverse strand), does it mean they use an **unstranded protocol ?**
- If this is indeed unstranded protocol, cufflinks requires the **XS flag**, which was not included when the STAR alignment was performed (it requires an additional command line parameter). Did the BIPAA team not use this parameter on purpose ? Am I missing something ?

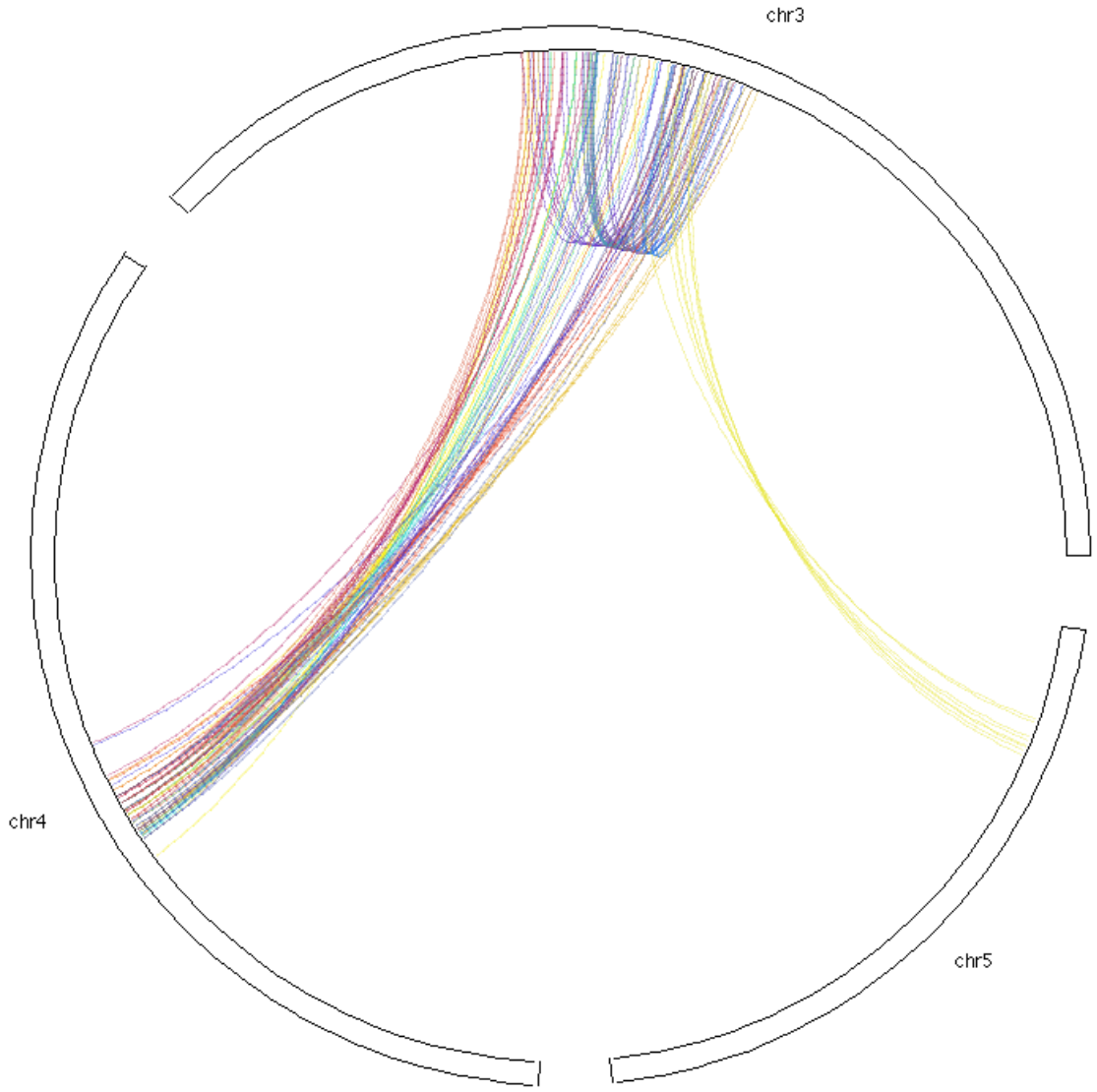

Figure 14.2: Circlular plot showing collinearity blocks between the 3 chromosomes containing CSD candidate hits in the GWAS. Gene coordinates are computational predictions from maker.

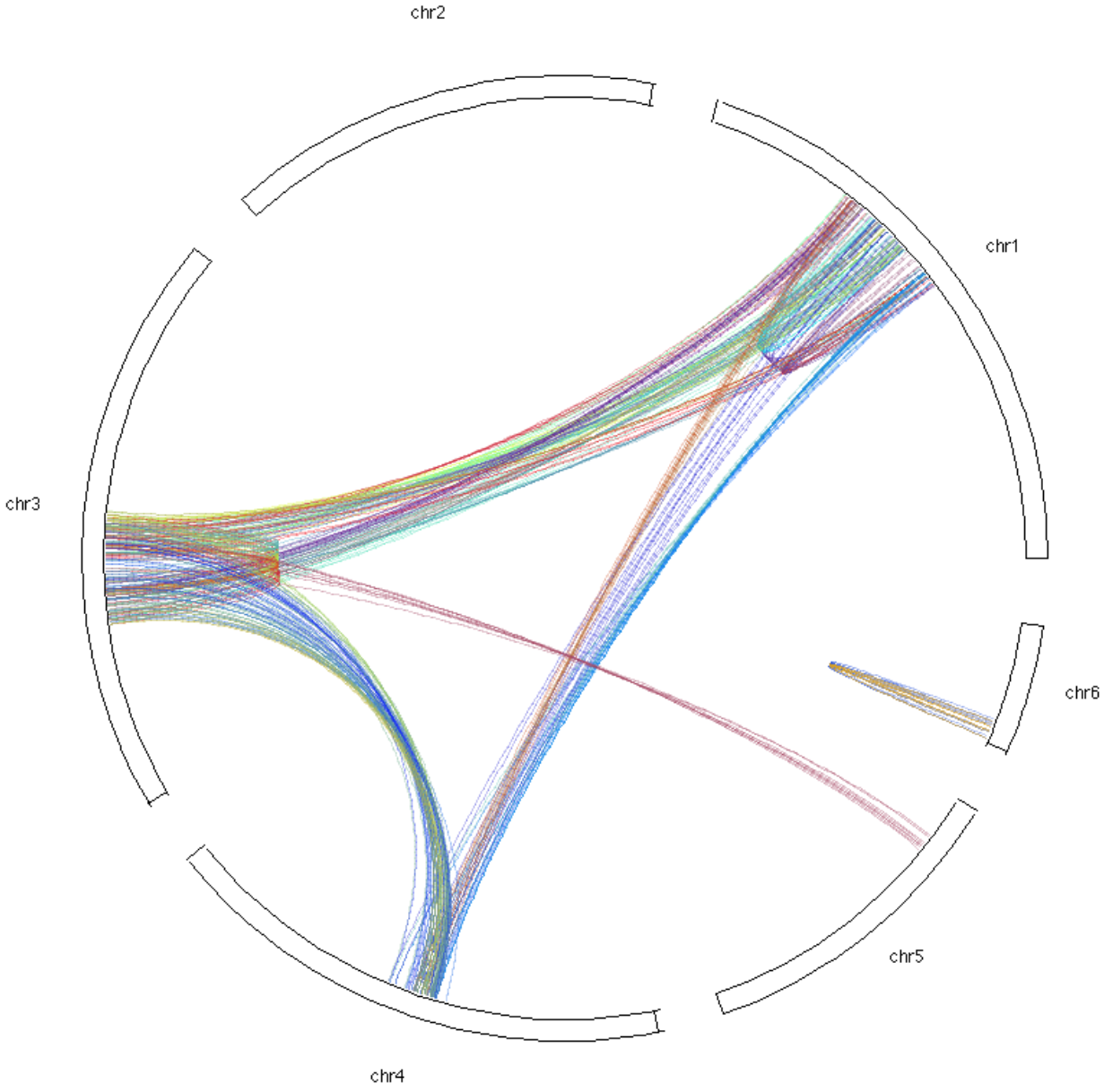

Figure 14.3: Circlular plot showing collinearity blocks between the all 6 chromosomes. Gene coordinates are computational predictions from maker.

Update: 25.11.2017 Issues resolved, larvae are from a stranded library and adults from an unstranded one. Both are single end. I cannot differentiate males and females because all individuals are pooled together without barcodes.

14.3.1 Data processing

I assembled the transcriptome from the reads alignment (BAM) file. The alignment was constructed with STAR and I assembled the transcriptome with cufflinks. I then extracted the transcripts coordinates and worked with these to identify gene coordinates.
Collinearity analyses were performed again with MCScanX using the merged tracks from maker genes and cufflinks transcripts to have a more complete set of genes.

14.4 gene homology: continued

14.4.1 visualisation

To combine the different layers of informations I have obtained, I produce circular plots using circos. The aforementioned layers include: Association mapping results for CSD, recombination rates (estimated from homozygosity relative to mother), MCScanX collinearity and the original contigs that were anchored into chromosomes.

Circos allows to integrate all these infos into a single plot, thus we can see whether CSD candidate regions show homologies with each other, if they are close to centromeres and if the contigs where they are found are small.

14.4.2 Collinearity parameters

Because the genome is highly fragmented and uncomlete (53% of length anchored to chromosomes), collinearity blocks will often be interrupted. We can also not be sure the CSD locus originate from a genomic duplication, and it might therefore not show collinearity. Changing MCScanX parameters (reducing minimum block size from 5 to 3 and varying the maximum gap between 10 and 30) did not improve the results much.

14.4.3 BLAST hits

To circumvent this issue, I tried to use direct blast hits between transcripts to build the links in the circos plot. I chose all **interchromosomal** significant blast hits using combined tracks of maker genes and cufflinks transcripts.

The combination of genome fragmentation and low rate of anchoring might pose a real issur to detect gene homologies in CSD regions.

Chapter 15

Sequence homology

In this chapter, I try to work around the drawbacks of genome incompleteness and unanchored contigs by measuring sequence homology and not only gene. There are several possible approaches and I will list those I can think of so that we can filter them later and find the most appropriate.
The different approaches will combine the following elements and ideas in different ways and orders.

- Whole genome alignment using minimap2.
- List of transformer orthologs from different hymenoptera species from orthoDB to identify its presence in the genome more reliably.
- Use original illumina reads from the assembly procedure to measure coverage and identify duplicated genes.
- list of single copy orthologs in hymenoptera/arthropods to have a coverage baseline
- During the assembly, duplicated genes are likely cut-off into single contigs. Use the paired end info from illumina reads to identify which are the neighbouring contigs of these genes.

15.1 Neighbours first

Strength: No expectations, can discover new stuff Weakness: Not very powerful, feature of interest will be missed if not in 1st neighbour

1. identify contigs neighbouring CSD region using pair end info
2. use original assembly reads to measure coverage of single copy orthologous genes in arthropods and compare it to genes in CSD contig and its neighbours to assess number of copies.
3. if N copies of a contig is more than 1, retrieve other neighbours in the assembly graph using pair end.
4. do these neighbours include other CSD regions ?
5. what is the concerned duplicated feature (*transformer* homologs) or other ?

15.2 transformer assumption

Strength: will likely yield results if a transformer homolog is responsible for CSD Weakness: Will be useless in finding any other factor

1. Look for transformer homologies in genome
2. do those homologies overlap CSD contigs ?
3. use single copy orthologs and original reads to assess number of copies of transformer homolog.
4. use pairs to find neighbouring contifs
5. do neighbouring contigs include CSD ones ?

15.3 Self-alignment

Strength: Assumption free Weakness: Will miss duplicated genes merged into single contig

1. Whole genome alignment using minimap2
2. find all contigs with multiple alignments.
3. do they include CSD contigs ?
4. retrieve their neighbouring contigs. Do they include CSD contigs ?

Chapter 16

Validating association results

16.1 reconsidering set of SNPs

Following Casper suggestions and comments, I tried running the association mapping procedure again without removing SNPs that are homozygous or missing in mothers from their family. This proved very useful, by allowing to keep many more SNPs in the analysis and it might be wiser to keep them as they should pose no harm to the analysis. The only drawback of including them is that CSD loci that are only heterozygous in few mothers will have a weaker signal.
From now on, I will be considering all SNPs for the association mapping. For the centromere identification however, I need to keep the aforementioned SNPs out as the metric used is heterozygosity relative to mothers.

16.2 identifying false positives

We can use the assumptions of the CSD model to identify false positive CSD hits. This can be done by counting the number of females that are heterozygous at 0, 1, 2,...,N of the N CSD candidates. If a candidate is actually a false positive, there should be no (or very few) females that are heterozygous at it, but at no other CSD locus.

16.3 Estimating genotype miscall rate

I am using unlikely mother-homozygous to offspring-heterozygous tansitions to obtain an estimate of the miscall rate.

16.4 sequentially dropping loci

To estimate the "importance" of csd loci, we can measure the number of females that are homozygous at all CSD loci, and repeat the operation after sequentially dropping different loci. If a loci is not really CSD, the count should not be strongly affected when we drop it.

16.5 pushing parameters

Low priority, on hold To minimize the part of the effect that is due to differing number of individuals, I will try increasing the minimum proportion of samples in which SNPs must be found (currently 0.8). I will also try with a coverage of 10, to have more confident calls.

Dropped	Hom. Females
0	7
1	13
2	19
3	10
4	21
1+2	31
1+3	18
1+4	38
2+3	24
2+4	48
3+4	29
1+2+3	44
1+2+4	97
1+3+4	53
2+3+4	62

Table 16.1: Number of females homozygous at all CSD loci after sequentially dropping loci.

Chapter 17

Comparison with bee CSD players

17.1 Feminizer

I retrieved the feminizer protein sequence for 8 Hymenopteran species, including bees and wasps. These sequences were retrieved from UniprotKB using a BLAST search from the *Nasonia vitripennis* feminizer. The list of gene IDs is:

- B3VN92 (Nasonia vitripennis)
- D9MZ89 (Euglossa hemichlora)
- B4Y115 (Bombus terrestris)
- B4XU23 (Melipona compressipes)
- A0A0H4URN0 (Apis florea)
- B1NW84 (Apis cerana)
- Q6V5L4 (Apis mellifera)
- B1NW85 (Apis dorsata)

I tblastn'd these protein sequences against the l.fabarum genome.

Chapter 18

TO DO

- Measure error rates (recombinations between SNP and CSD) using number of males that are heterozygous at each CSD peak
- Number of females homozygous at all CSD loci by consecutively dropping loci to assess "importance" of locus.
- Cleanly annotate transformer using apollo and reads from RNA-seq
- identify over represented repeats (centromeric repeats) and delimit centromeres from the region they span
- assemblers tend to put duplicated genes in separate contig (multiple graph paths going through it). Use original Illumina paired end reads to retrieve unanchored contigs neighbouring anchored contigs: potentially recovering multiple occurrences of CSD duplicated feature (if any)
- get centimorgans coordinates from bp using linkage map and place centrosomes.
- look at the proportion of recombination for different GWAS hits: does it match the mode (assuming single mode) in the prop males among 2n offspring plot.
- compute prop.het loci in centromeric window for all individuals and isolate TFA based on distribution if bimodal.
- try above with different values for window sizes and select based on number of loci contained in window.
- Compute coverage along genome for all loci including monomorphic ones.
- Isolate regions of abnodmally high coverage, eventually exclude them. Watch out for overmerging of paralogs and repetitive sequences.
- Visualize average heterozygosity (or allele frequency) at each SNPs in offpring versus in mother to assess transmission bias (hom SNPs in mothers more often het in offspring or reversse).
- make snp calling less reliant on correct mother calls and exclude SNPs only if mother is hom/missing AND less than N offspring are heterozygous (small N)
- automate threshold selection for inferring ploidy