**2025 International Statistical Genetics Workshop - Genotype QC**

In this practical, we'll work through some of the quality control steps involved in cleaning genotype data prior to running GWAS.

The practical has been split into 6 main steps
**Step 1**. Data and Formats
**Step 2.** Check for reported/genotype sex discrepancies
**Step 3**. Obtain information on individuals missing SNP data
**Step 4**. SNP QC: SNPs missing data; MAF; Hardy-Weinberg
**Step 5**. Sample QC: genotype call rate and heterozygosity
**Step 6.** Checking for relatedness

**Step 1. Data and Formats**

**1.1 Navigating to the practical workspace**
To get started, ssh into the workshop server.
1) Go to: <https://workshop.colorado.edu/ssh/>
2) Enter workshop login ID and password

The content for all of the practicals can be found in your home directory under practicals ~/practicals
Navigate to the directory for this practical session
cd /practicals/1.2.DataGeneration+QC\_AleshaHatton/final
We are going to create a directory to work from in this practical. This will help to maintain a clean working space.
mkdir QCdata
cd QCdata

Most of the practical will be run using the ssh terminal. For some steps we will be using R so we can visualise the data. Unless indicated otherwise, run the commands provide in the terminal.

For many of the steps in the practical we will be using plink. There are many online resources to help you navigate the software such as <https://www.cog-genomics.org/plink/1.9/>

**1.2 Looking at the genetic data**
The genotype data has been provided in plink txt format.
There are two files (.ped and .map)

The .ped (pedigree format) file holds both phenotype and genotype information. Each row is an individual.

The .ped file contain the following columns:

|  |  |
| --- | --- |
| **FID** |  Family ID |
| **IID** |  Individual ID |
| **PID** |  Paternal ID |
| **MID** |  Maternal ID |
| **SEX** | Sex is coded with Male = 1, Female = 2HINT: think of number of X chromosomes |
| **Phenotype** | For case-controls: Control = 1; Case = 2; Missing = 0/-9 |
| **SNP data** | SNP data is represented with 2 columns, one for each allele.SNPs are coded either with letters (a,c,t,g) or numbers (1,2,3,4) |

Take a look at the .ped file
less -S /usr/local/data/cc.ped

The -S flag to stop rows from wrapping. Use q to quit. Scroll down to get an idea of how the trait is coded.

You can check the minimum and maximum value of the trait (column $6) with the following command:
awk 'NR==1{min=max=$6};$6<min{min=$6};$6>max{max=$6}END{print min, max}' /usr/local/data/cc.ped

If you're certain it's not continuous, then you get more of an idea about sample size and missing data with:
awk '{print $6}' /usr/local/data/cc.ped | sort -n | uniq -c

**Question 1a:**How is the trait coded?

**Answer 1a:**The traits is coded as 1 (controls) and 2 (cases).

Look at the .map file
The .map file contains information about the genetic variants. Each row is a genetic variant.
The.map file contain the following columns:

|  |  |
| --- | --- |
| **CHR** | Chromosome |
| **RS** | Reference SNP ID |
| **Distance** | Genetic Distance (in centimorgans)Often there are zeros are in the distance column = unknown |
| **Position** | Human genome reference base position |

 less /usr/local/data/cc.map

**The basics of plink**
A basic PLINK command follows this general structure:
plink --file input\_filename --command --output output\_filename

Breakdown:
plink – The command that calls the PLINK program.
--file – load in the dataset
--command – provides plink with an instruction (some commands also need input values e.g. thresholds)
--output – instruct plink what to name the output

**1.3 Convert to PLINK binary file format**
Next we want to convert the ped & map files to plink bfile (binary) formatted files. It is faster to recall the data in binary format and requires less space for storage

The plink command --make-bed creates the binary format files:

* fam = information on the pedigree and phenotype
* bim = this is the map file with the alleles for each variant
* bed = binary file with each allele for each person

Each time we remove either variants or individuals we will create a new set of bfiles using the --make-bed flag.
plink --ped /usr/local/data/cc.ped --map /usr/local/data/cc.map --make-bed --out cc.begin

NOTE: Errors in plink will stop progress. Warnings are to inform your decisions.

What about the .hh file?
Warning: 2177 het. haploid genotypes present (see cc.begin.hh)
Heterozygous haploid errors result from heterozygous calls on chromosome X for males. Males have one X chromosome. They are hemizygous. They ought not be heterozygous for loci on chromosome X, unless the loci in are in pseudo-autosomal regions. We can use the split-x command to check, which we will do later in the tutorial. Heterozygous haploid errors are saved into files with the suffix .hh if you want to examine them in more detail.

**Question 1b:** How many SNPs, people, females, and controls are there?

**Answer 1b:**There are: 484128 SNPs, 193 people, 86 females, 66 controls.

**Step 2. Check for reported/genotype sex discrepancies**
This step is designed to check for a mismatch between sex reported in the ped file and sex from genotype data. This can help identify if IDs have been mismatched to genetic data. This is uncommon with modern machinery and barcoding, but still possible.

**2.1 Running PLINK sex-check**

Since we are only checking details we will use the PLINK --out command.

plink --bfile cc.begin --check-sex --out sex

**2.2 Viewing discrepant samples**

less sex.sexcheck

This file has a column that indicates the reported sex and sex inferred from the genetic data.  --check-sex is a heterozygosity check of chromosome X. Males have one chromosome X, females have two. We expect heterozygosity on chromosome X for females but not males. The F statistic (inbreeding coefficient) based on SNP homozygosity. For males it should be close to 1, for females it should be close to 0. Typically, above .8 is classed as male, below .2 is classed as female. The STATUS column (OK/PROBLEM) reflects if reported sex matches the genetic data.

Select the individuals flagged as not matching
grep PROBLEM sex.sexcheck

for a cleaner output, use this command
grep 'PROBLEM\|FID' sex.sexcheck | column -t

"\|" operator looks for multiple matches, and \| treats the | as an operator instead of a string. column -t tab-separates the output

Copy discordant samples into a file to remove from the data set.
grep PROBLEM sex.sexcheck > sex.drop

**Question 2a: How many mismatches were flagged?**

**Answer 2a**: 3 samples were flagged as sex mismatches

**2.3 Plotting X chromosome heterozygosity in R**
Next we are going to plot the sample call rate in R using Rstudio. All the Rcode is available in a pre-prepared script
Using Rstudio:
1) Open a separate browser window
2) Go to <https://workshop.colorado.edu/rstudio> (open in a new tab)
3) Login with your username and password
4) In the "Console" tab, navigate to your working directory

setwd("~/practicals/1.2.DataGeneration+QC\_AleshaHatton/final/")

5) To open the Rscript, go to File -> Open -> QC\_steps.R
6) Read through the R script through to section 2.3

R code: Plotting X chromosome heterozygosity
Read in PLINK sexcheck file
sex <- read.table("sex.sexcheck", header=T)
View contents
head(sex)
Histogram
hist(sex$F, xlab="Probability chrX alleles are from a single ancestor", main = "chrX Heterozygosity")
Boxplot split by reported sex
boxplot(split(sex$F, sex$PEDSEX), main = "Checking reported/genotype sex discrepancies", xlab="Reported sex", ylab="chrX Heterozygosity")
Can you identify on the plot which samples are potential mismatches?
(Keep the Rstudio window open as we will use it for later steps)

**2.4 Remove  sex-check discrepant samples**
Navigate back to the ssh terminal
Remove the sex-discrepant samples from the data set
plink --bfile cc.begin --remove sex.drop --set-hh-missing --make-bed --out cc.qc1

NOTE: the --set-hh-missing command is explained in more detail later in the practical (extended QC steps)

**Step 3: Obtain information on individuals missing SNP data**

**3.1 Running PLINK --missing**
The --missing command will give output files on both types of missingness:

* individuals missing SNP data -> .imiss
* variants missing data from individuals -> .lmiss (NOTE: the "l" stands for locus)

 plink --bfile cc.qc1 --missing --out miss
 Have a look at both:
less miss.imiss
less miss.lmiss

**3.2 Plotting sample call rate in R**
Sample Call Rate: individuals missing genotyped data
Read in PLINK imiss file
missI <- read.table(file="miss.imiss", header=TRUE)

View contents
head(missI)

Histogram of F\_MISS
h1 = hist(missI$F\_MISS, breaks=seq(0,.1,.005))
c1 = ifelse(h1$breaks<0.05, "grey", "red")[-length(h1$breaks)]
plot(h1, col=c1, xlab="Proportion of missing data: individuals", main ='Sample Call Rate')

Look at highest missing SNP
frequency max(missI$F\_MISS)

Look at top 20 samples with the highest missing SNP frequency
head(missI[order(missI$F\_MISS,decreasing=T),],20)

# Check how many individuals have a call rate less than .95 (or 5% missing genetic data)
sum(missI$F\_MISS > .05)

We could exclude these individuals here. However, many samples with missing genotype calls can be due to many poorly genotyped snps. It can be a good idea to can check the quality of the SNP data first

**Question 3a:**How many individuals will we drop if we keep a sample call rate of >=95 (i.e. remove those with >5% missing SNPs)?

**Answer 3a** 36 samples

**Step 4: SNP QC: SNPs missing data; MAF; Hardy-Weinberg**
**4.1 Plot SNP call rate in R**
We are going to plot the genotype call rate to see how many SNPs are missing information from too many people
read in PLINK lmiss file
missL <- read.table(file="miss.lmiss", header=TRUE)

view contents
head(missL)

Histogram of F\_MISS
h2 = hist(missL$F\_MISS, breaks=seq(0,.4,.005))
c2 = ifelse(h2$breaks <- 0.05, "grey", "red")[-length(h2$breaks)]
plot(h2, col=c2, xlab="Proportion of missing data: SNPs", main ='Genotype Call Rate')

How many SNPs are missing more than 5% of data?
sum(missL$F\_MISS > .05)

NOTE: It is best to be conservative with SNP call rate filtering, as imputation will provide many more SNPs than those lost here. Our imputation will be better quality if we restrict to the highest quality SNPs for imputation. We want to maximize our sample size at this point. So we will clean out the poor quality SNPs first, and then individuals later

**4.2 Filter SNP call rate < 95% with PLINK --geno**

Remove variants that are missing information from too many individuals. We will keep those with a genotype call rate of at least .95 (i.e. variants with data from at least 95% of our sample).

plink --bfile cc.qc1 --geno 0.05 --make-bed --out cc.qc2

**Question 4a:**How many variants were removed by the call rate filter?

**Answer 4a**: 127,030 variants were removed

**4.3 Case-control call rate difference with PLINK --test-missing**
When cleaning case-control data, check that variants are not disproportionately missing between cases and controls.
plink --bfile cc.qc2 --test-missing --out case-control

**4.4 Filter case-control call rate SNPs with p < 1e-5**
This awk code will grab any variants that differ in missingness between cases and controls with a p value < .00001 and save those variants into a file to be dropped.
awk '$5<=0.00001' case-control.missing > case-control.missing.drop
less case-control.missing.drop

 There are none in this sample, as the file is empty. If there were, then we could use the PLINK --remove command. (We will implement --remove later in this practical.)

**4.5 Running PLINK --freq**
Obtain MAF information on the whole sample so we can plot the distribution of MAF.
plink --bfile cc.qc2 --freq --out maf

NOTE: When using bfiles, PLINK traditionally sets the minor allele as A1. PLINK2 no longer does this.
PRO TIP: using PLINK's '--keep-allele-order' command will prevent PLINK from setting the minor allele as A1. This is useful if you import VCF data and want A1/A2 to reflect REF/ALT

**4.6 Plot the sample allele frequency spectrum in R**
Read in PLINK .frq file
maf <- read.table("maf.frq", header =TRUE)

View contents
head(maf)

Histogram
h3 = hist(maf$MAF, breaks=seq(0,.5,.005))
c3 = ifelse(h3$breaks>0.01, "grey", "red")[-length(h3$breaks)]
plot(h3, col=c3, xlab="Minor allele frequency", main ='MAF')

Number of variants with a MAF < 0.01
sum(maf$MAF < 0.01)

**4.7 Filter low-frequency SNPs with PLINK --geno**
plink --bfile cc.qc2 --maf 0.01 --make-bed --out cc.qc3

**Question 4b:**How many variants were removed using the MAF filter?
NOTE: The minor allele count for a 0.01 MAF SNP in our sample
MAF \* (NCHROM) or 0.01 \* (193\*2) = 3.86 minor alleles
As sample sizes get larger, this QC threshold should get lower for GWAS studies

**Answer 4b**: 50,113 variants were removed

**4.8 Running PLINK --hardy and Filter SNPs with --hwe 1e-6**

Check Hardy-Weinberg Equation (HWE) deviation and remove variants with P < 1e-6.
With case-control data, the default HWE check only conducts the check on the controls. We have overwritten that here with the --include-nonctrl flag.
The --hardy command will give us the HWE values for each SNP calculated on the whole sample, the cases, and the controls.
The --hwe command will drop those variants that are below threshold.

plink --bfile cc.qc3 --hardy midp --hwe 1e-6 midp include-nonctrl --make-bed --out cc.qc4

NOTE The 'midp' is a p-value adjustment added to test to bring the null rejection rate in line with the nominal p-value and also reduces the filter's tendency to favor retention of variants with missing data (see <https://www.cog-genomics.org/plink/1.9/filter#hwe>)

**Q4c**. How many variants were dropped due to HWE?

**Answer 4c**: 636 variants

Given the warning about variation due to chromosome X, let's check if these HWE errors are all on chromosome X (coded as 23 in PLINK).We can do this by keeping those SNPs with P < 1e-6

grep CHR cc.qc4.hwe > hwe
awk '$9<1e-6' cc.qc4.hwe | grep ALL >> hwe

Sort and count number of variants on each chromosome.
awk '{print $1}' hwe | sort -n | uniq -c

Not specific to chromosome X, as there were variants on all chromosomes dropped due to HWE deviations from the expected.

**Step 5. Sample QC: genotype call rate and heterozygosity**
**5.1 Filtering samples with < 95% call rate with PLINK --mind**
plink --bfile cc.qc4 --mind 0.05 --make-bed --out cc.qc5

**Question 5a:**How many samples were dropped with call rate < 0.95?
Compare this to how many we would have dropped if we removed individuals before cleaning up the variants (above in Q3a).

**Answer 5a**: 3 samples were removed with call rate < 0.95.

Cleaning up the variants before checking individuals, means we removed only 1.5% of this sample instead of 20% of this sample if we had done the reverse.

**5.2 Comparing pre and post QC call rates**

Lets obtain missingingness information from the data file created at the end of step 4 when we finished cleaning variants (Post QC).
plink --bfile cc.qc4 --missing --out miss2

**Plotting pre and post QC call rates in R**
Read in PLINK .imiss file
miss2I <- read.table(file="miss2.imiss", header=TRUE)

View contents
head(miss2I)

Histogram
h4 = hist(miss2I$F\_MISS, breaks=seq(0,.1,.005))
c4=ifelse(h4$breaks<0.05, "grey", "red")[-length(h4$breaks)]
par(mfrow=c(2,1)) ## Plot both graphs
plot(h4, col=c4, xlab="Proportion of missing data: individuals", main ='AFTER QC')

Compare with the plot from before cleaning the variants
plot(h1, col=c1, xlab="Proportion of missing data: individuals", main ='BEFORE QC')

par(mfrow=c(1,1)) ## reset plotting

Check how many individuals have a call rate less than .95 after variant QC (or 5% missing genetic data)
sum(miss2I$F\_MISS > .05)

**5.3 Per-sample Heterozygosity PINK --het**
Obtain the heterozygosity information on the autosomes (chr 1-22).  Too much heterozygosity might be a problem from DNA contamination. Allele frequencies are used in this calculation. Ideally, the sample is not small. Since ours is, we use the small-sample modifier.

plink --bfile cc.qc5 --het small-sample --out het

**5.4 Filtering samples with excess / depleted heterozygosity**
Calculate the proportion of heterozygosity:
(total genotypes - homozygotes) / total number of autosomal genotypes
echo "FID IID obs\_HOM N\_SNPs prop\_HET" > het.txt
awk 'NR>1{print $1,$2,$3,$5,($5-$3)/$5}' het.het >> het.txt

This awk command will calculate the values of the proportion of het that corresponds to +3SD and -3SD around the mean(on prop\_HET, the 5th column of a file that we have just created)
awk 'NR>1{sum+=$5;sq+=$5^2}END{avg=sum/(NR-1);print avg-3\*(sqrt(sq/(NR-2)-2\*avg\*(sum/(NR-2))+(((NR-1)\*(avg^2))/(NR-2)))),avg+3\*(sqrt(sq/(NR-2)-2\*avg\*(sum/(NR-2))+(((NR-1)\*(avg^2))/(NR-2))))}' het.txt

Save individuals with heterozygosity rates >3SD into a file to then drop them.
awk '$5<=0.299582 || $5>= 0.326536' het.txt> het.drop

**Question 5b.** How many samples were >3SD from the mean heterozygosity rate?

**Answer 5b**: 3 samples

**Step 6. Sample QC: Checking for cryptic relatedness**

We estimate relatedness between every pair of individuals using a genetic relationship matrix (GRM). This is a (n x n) matrix (n being the number of individuals) where the values correspond to the average correlation between each pair of individuals across genotyped snps. Pairs containing different individuals are in the off-diagonal elements while diagonals represent only one individual (and will therefore have a value close to 1).

The genetic relationship between two individuals can be estimated by the following equation:

([Yang et.al., 2011](https://pmc.ncbi.nlm.nih.gov/articles/PMC3014363/))

**6.1 Calculate genetic relatedness using a genetic relationship matrix (GRM)**
We are going to calculate relatedness in plink
plink --bfile cc.clean --make-grm-gz no-gz --out cc.rel

This command calculated relatedness but displays the output as a list instead of a matrix (making it easier to view).

Look at the output
less cc.rel.grm

Note: the ID's were written to cc.clean.grm.id. The values in the first two columns of cc.rel.grm correspond to the order the individuals were in the file, not their .fam IDs!

Do any individuals have a relatedness larger than 0.05? (remember diagonals represent only one individual)
awk '$4>0.05 && $1!=$2 {print}' cc.rel.grm

Typically, we remove one of a pair of individuals with estimated relatedness larger than the specified cut-off value (e.g. 0.05).  Note: We selectively remove individuals to maximize the remaining sample size rather than doing it at random.

plink --bfile cc.clean --rel-cutoff 0.05 --out cc.unrelated

This is the end of the main QC section. There are additional steps if you want to keep going!

**# === Main QC === #**
Step 1. Data and Formats
Step 2. Check for reported/genotype sex discrepancies
Step 3. Obtain information on individuals missing SNP data
Step 4. Variant QC: SNPs missing data; MAF; Hardy-Weinberg
Step 5. Sample QC: genotype call rate and heterozygosity
Step 6. Checking for cryptic relatedness

**## === Additional QC ===#**
Step 7. Genome Build checking
Step 8. Haploid het checking / filtering
Step 9. HapMap3 reference panel merging and strand ambiguous SNP checking / filtering
Step 10. Looking at additional QC pipelines
Step 11. Looking at imputation pipelines
Step 12. Clean up your workspace

**If you want to keep working through the additional QC steps then dont clean up your workspace just yet! (instead click next)**

**If you are finishing here then it can be a good idea to clean up your workspace**
It is normal to create many PLINK datasets in QC, so its important to clean it up!

Look at how much space you've taken up
ls -lh | head -n 1

Use the \* to find and remove specific files

E.g. all files starting with cc.clean
ls -lh cc.clean\*

All files ending in bed
ls -lh \*bed

remove only qc PLINK .bed/.bim/.fam/.hh files
ls -lh cc.qc\*
rm cc.qc\*bed
rm cc.qc\*bim
rm cc.qc\*fam
rm cc.qc\*hh
ls -lh cc.qc\*

when you are done with the practical, erase all files in the folder (be careful with this command!)

rm ~/practicals/1.2.DataGeneration+QC\_AleshaHatton/final/QCdata/\*

Remember all files can be re-created by running the commands in this workbook.
We have also provided all the commands in the .sh and .R scripts in the practical directory

**Step 7. Genome Build checking**
Look at your .map or .bim file

We are going to cross reference the base position of several SNPs with what is shown in the genome browser for several builds.
Navigate to the genome browser: [https://genome.ucsc.edu](https://genome.ucsc.edu/)
Click on Genomes
Under Human Assembly use the drop-down menu to go to a build (e.g.??)

Enter an rs# or chr:bp (e.g. rs3094315 or chr1:752566)
Click Go
Click on one of the links for dbSNP i.e.
Common (1000 Genomes Phase 3 MAF >= 1%) Short Genetic Variants from dbSNP Release 153 or Simple Nucleotide Polymorphisms (dbSNP build XXX)
There are a lot of possible links here for different releases of dbSNP.

Then click on the rs# link on the left-hand side
NOTE: This is inside the graph that describes each UCSC track being plotted, NOT the lists below! #

Here you will see useful information about the variant including position, alleles, frequency in from different projects.

You are looking to see if the position listed against the rs# in the map file matches the position in the build. As builds are updated, the position number for some of the rs# will change. Check a couple of rs# across different builds.

**Q7**. Which build are these data mapped to?

To LiftOver your genomic positions to a different build, see <https://genome.ucsc.edu/cgi-bin/hgLiftOver>
<https://hail.is/docs/0.2/functions/genetics.html#hail.expr.functions.liftover>

**Step 8. Haploid het checking / filtering**

When we removed samples with sex-check discrepancies, we still had heterozygous haploid genotypes present.

**8.1 Using PLINK --split-x to evaluate the X/Y psuedo-autosomal region**

Use split-x to see if the remaining issues are due to the pseudo-autosomal regions. This command is taking the variant position boundaries of the pseudo-autosomal region according to build 37 (hg19) and changing the chromosome codes of all variants  in the region to XY. So plink will not treat these regions as haploid. If the data was on a different build, you would need to check if plink will implement the appropriate flag

plink --bfile cc.begin --split-x b37 no-fail --make-bed --out cc.splitx

**Q8a**. Did this remove the remaining het haploid genotypes?

**8.2 Why we used PLINK --set-hh-missing**
These remaining het haplod issues do not appear to be from pseudo-autosomal regions. Therefore, these remaining variants would be removed for analytical commands, but we can set them to missing.

This is why the '--set-hh-missing' is used to create the cc.clean PLINK files

NOTE: the number of variants and individuals did not change. The errors come from specific variants for certain people. Only those errors are set to missing. If you have time (and want to) you can explore the details of which variants and people in the \*.hh files.

**Step 9. HapMap3 reference panel merging and strand ambiguous SNP checking / filtering**

**9.1 Selecting strand ambiguous SNPs**

First drop strand ambiguous alleles from our cleaned data set and create a file with a list of our variants (rs#).

NOTE on strand ambiguous alleles: Each chromosome is double-stranded DNA. The two strands bond together with A paired to T and C paired to G. Strands may be named as + and - (or forward and backward, or top and bottom) but with genotyped data this naming is quite arbitrary. If we are combining data across platforms or samples there is uncertainty around if the strand reference is the same. Often we get around this uncertainty by removing strand ambiguous alleles.

In the 5th and 6th column of the bim file, strand ambiguous SNPs will have an A and T as the two possible alleles for that variant, or they will have a C and G as the two possible alleles.

We will remove all strand ambiguous allele SNPs before merging our data with HapMap3 (HM3). This bit of code selects the possible strand ambiguous allele combinations # from columns 5 and 6, and then prints the RS# along with the word ambig. Those variants not selected just have the RS# printed
Then we select those rows of data without the word ambig and copy them into a file cc.clean.snplist

awk '{if(($5=="T" && $6=="A")||($5=="A" && $6=="T")||($5=="C" && $6=="G")||($5=="G" && $6=="C")) print $2, "ambig"; else print $2}' cc.clean.bim | grep -v ambig > cc.clean.snplist

**9.2 Extract HapMap3 (HM3) data on our remaining SNPs**

plink --bfile /usr/local/data/HM3 --extract cc.clean.snplist --chr 1-22 --make-bed --out hm3-oursnps

Obtain a list of HM3 snps

awk '{print $2}' /usr/local/data/HM3.bim > HM3.snplist

**9.3 Reduce our data to only include SNPs also in HM3**

plink --bfile cc.clean --chr 1-22 --extract HM3.snplist --make-bed --out cc.clean-hm3snps

**9.4 Merge our cleaned data and the HM3 data (for the snps that matched)**

plink --bfile cc.clean-hm3snps --bmerge hm3-oursnps.bed hm3-oursnps.bim hm3-oursnps.fam --make-bed --out cc.merged

This gives an error!
PLINK suggests it may be due to strand inconsistency and provides an output file with the potential mismatched SNPs
e.g. rs10000037 T C in our data and A G in HapMap.

If we flip the strand of the mismatched SNPs in our data the T flips to A and the C flips to G and our rs10000037 data will align with HapMap.

**9.5 Flip the strand for the missnps in our data using PLINK --flip**

plink --bfile cc.clean-hm3snps --flip cc.merged-merge.missnp --make-bed --out cc.flipped

**9.6 Merge again, using the file with the flipped SNPs**

plink --bfile cc.flipped --bmerge hm3-oursnps.bed hm3-oursnps.bim hm3-oursnps.fam --make-bed --out cc.flipped.merged

**Step 10.1. Looking at additional QC pipelines**
Visit <https://sites.google.com/a/broadinstitute.org/ricopili/>

**10.1 Look at the Pre-imputation (QC) page, and read the Input requirements and Technical Details.**

Many of the details automate the commands that you have just run through in PLINK

Scroll the PCA page to see how this module:
 - filters and LD prunes genotype data
 - checks relatedness
 - produces PCA graphs

Custom installation of Ricopili to any high performance cluster computer is detailed here:  <https://docs.google.com/document/d/14aa-oeT5hF541I8hHsDAL_42oyvlHRC5FWR7gir4xco/edit?usp=sharing>

Supported job scheduler environments:
- qsub
- bsub
- slurm
- Google Cloud Platform

**10.2 GWASpy pipeline**
Visit <https://gwaspy.readthedocs.io/en/latest/>
Written Lindo Nkambule, this pipeline uses Hail and Python to run parallel modules to Ricopili

Visit <https://github.com/atgu/hgdp_tgp/tree/master/tutorials#links-to-view-notebooks>
To check out the how it was used in the tutorial jupyter notebooks for the HGDP+1000genomes dataset
Supported job scheduler environments:
- Google Cloud Platform

**10.3 Separate GWAS tutorial on GitHub**
Visit <https://github.com/MareesAT/GWA_tutorial>
Written by Andries Marees, this is another full GWAS tutorial publicly available on GitHub

 **Step 11. Looking at imputation**
If you want to learn more about imputation, there is are great practical workshop sessions from ASHG20202 on the Michigan Imputation Server Site

Visit: <https://imputationserver.readthedocs.io/en/latest/workshops/ASHG2020/>

There are many factor that affect imputation including:

* Size of reference panel
* Density of genotyping array
* Minor allele frequency in the reference panel of variant being imputed
* Haplotype accuracy in reference and study samples
* Sequencing coverage of the reference panel
* Genetic similarity between reference panel and study samples
* Demographic history of the population