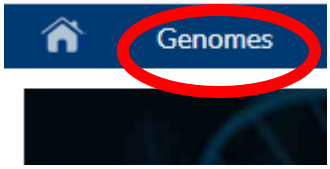
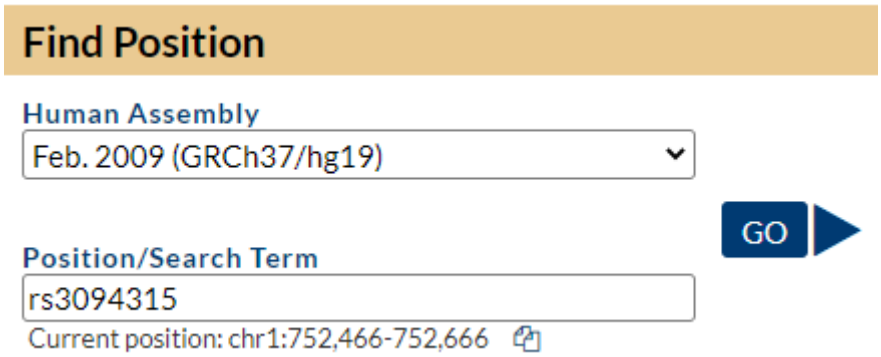
```
# The genetic data for today's tutorial comes from
http://labs.med.miami.edu/myers/LFuN/LFUN/DATA.html
# Meyers 2007 "A survey of genetic human cortical gene expression"
# The phenotypes are simulated.

# Resources
# https://www.cog-genomics.org/plink/1.9
# http://zzz.bwh.harvard.edu/plink/
# https://genome.ucsc.edu
# https://workshop.colorado.edu/rstudio

# Copy the files to your working area
mkdir QC
cd QC
cp /home/katrina/2021/QC .

# Check you have the required files:
# cc.ped
# cc.map
# qc-plots.R
# HM3.bed
# HM3.bim
# HM3.fam

# Answers to the questions are at the end.

#~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
STEP 1.  Data and Formats
#~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~

# We begin by looking at the data.

# The ped (pedigree format) file holds both phenotype
# and genotype information.
# Each row is an individual.
# Ped files always begin with:
# FamilyID IndividualID PaternalID MaternalID SEX
# Sex is coded with Male = 1, Female = 2
# HINT: think of number of X chromosomes
# Then follows the trait(s).
# For case-controls:
# U = 1 = controls; A = 2 = cases; X = 0 = -9 = missing
# Then come the columns for all the 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)

# 1.1
# Look in the ped file:
# Use the -S flag to stop rows from wrapping

less -S cc.ped

# REMINDER: q to quit

# Scroll down to get an idea of how the trait is coded.

# Q1a. How is the trait coded?
```

```
# You can check the minimum and maximum 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}'
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}' cc.ped | sort -n | uniq -c

# 1.2
# Look at the map file
# Columns: CHR RS# Distance Position
# Often zeros are in the distance column = unknown

less cc.map

# Q1b. Which build are these data mapped to?

# Cross reference the base position of several SNPs with
# what is shown in the genome browser for several builds.

# https://genome.ucsc.edu
# Click on 'Genomes'
```




```
# Under 'Human Assembly' use the drop-down menu to go to a build.
# Enter an rs# 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.
```

Common (1000 Genomes Phase 3 MAF >= 1%) Short Genetic Variants from dbSNP Release 153

rs3094315 at chr1:752466-752666

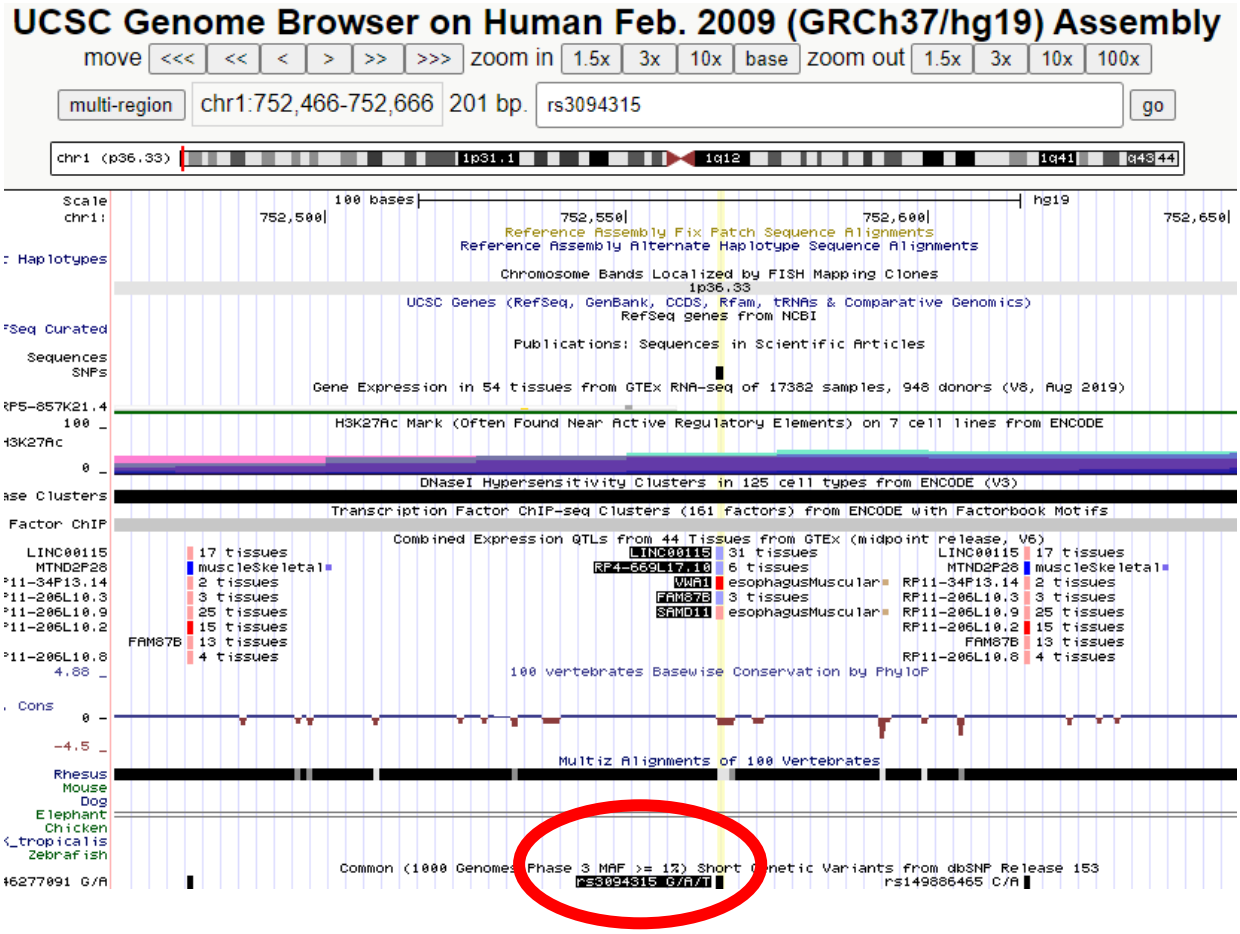
All Short Genetic Variants from dbSNP Release 153

rs3094315 at chr1:752466-752666

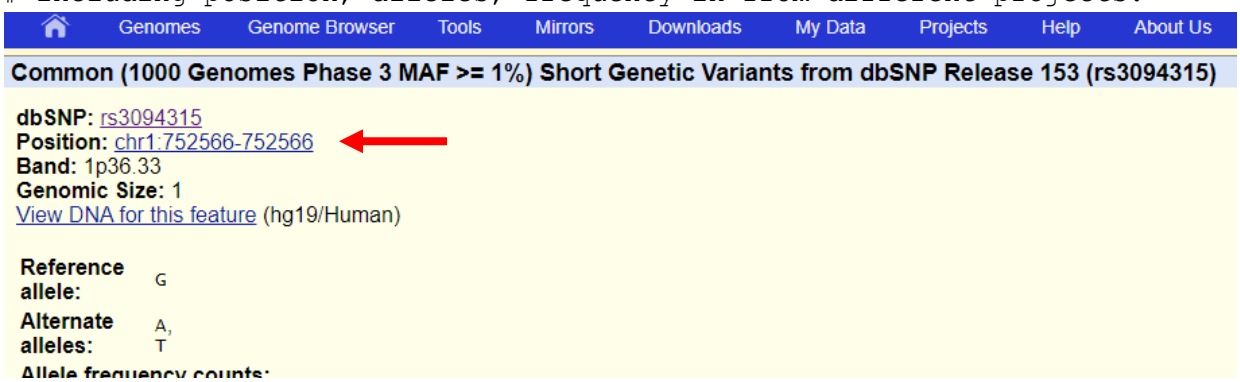
Simple Nucleotide Polymorphisms (dbSNP 151)

rs3094315 at chr1:752316-752816

Then click on the rs# link



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.

# 1.3
# Convert the ped & map files to plink bfile (binary) formatted files.

# 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 cc.ped --map cc.map --make-bed --out cc.begin

# Q1c. How many SNPs and people are there?

# What about this?
# "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.

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


#~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
STEP 2.  Check for sex errors
#~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~

# 2.1
# 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.
# Since we are only checking details we will use the PLINK --out command.

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

# 2.2
# Look at the created file

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 reflects if reported sex matches the genetic data.

# 2.3
# Select the individuals flagged as not matching

grep PROBLEM sex.sexcheck

# Q2a. How many mismatches were flagged?

# 2.4
# Plot chromosome X heterozygosity in R (use qc-plots.R script)

# We are checking for errors here to assess if there is a major problem.
# There are some reports that checking sex ought to be done after
# pruning the variants to be independent (in linkage equilibrium)
# So for now we will continue with all of our sample,
# and see if there is a problem when checking sex on independent variants
# later in the tutorial.


#~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
STEP 3.  Obtain information on individuals missing SNP data
#~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~

# 3.1
# 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.begin --missing --out miss

# Have a look at both: miss.imiss miss.lmiss

# 3.2
# Plot the Sample Call Rate in R

# Q3a. How many individuals will we drop if we keep a
# Sample Call Rate of >=95 (i.e. remove those with >5% missing SNPs)?


#~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
STEP 4.  Remove variants: SNPs missing data; MAF; Hardy-Weinberg
#~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~

# Now we start cleaning up our data!

# 4.1
# SNPs missing data.

# 4.1.1
# 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.begin --geno 0.05 --make-bed --out cc.qc1

# 4.1.2
# Plot the Genotyping Call Rate in R.

# Q4a. How many variants were removed?

# 4.1.3
# When cleaning case-control data, check that variants are
# not disproportionately missing between cases and controls.

plink --bfile cc.qc1 --test-missing --out case-control

# 4.1.4
# This awk code will copy 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

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

# 4.2
# Minor Allele Frequency.

# 4.2.1
# Obtain MAF information on the whole sample.
# We do not need to create this frequency file for QC,
# but it is a useful command to know.
# And by creating the file we can plot the distribution of MAF.

plink --bfile cc.begin --freq --out maf

# NOTE: When using bfiles, PLINK always has the minor allele as A1.
# Not all programs do this!

# 4.2.2
# Plot the MAF frequencies in R

# 4.2.3
# Remove variants with minor allele frequency (MAF) < .01

plink --bfile cc.qc1 --maf 0.01 --make-bed --out cc.qc2

# Q4b. How many variants were removed due to MAF?
# NOTE: There will be less than from the result in R as that
# was obtained prior to any QC steps.


# 4.3
# Hardy-Weinberg.

# 4.3.1
# Check Hardy-Weinberg Equation (HWE) deviation
```

```
# Remove variance 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.qc2 --hardy midp --hwe 1e-6 midp include-nonctrl --make-
bed --out cc.qc3

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

# 4.3.2
# Given the warning about variation due to chromosome X
# Let's check if these HWE errors are all on chromosome X.
# Keep those SNPs with P < 1e-6

grep CHR cc.qc3.hwe > hwe
awk '$9<1e-6' cc.qc3.hwe | grep ALL >> hwe

# Sort and count number of variants on each chromosome.

awk '{print $1}' hwe | sort -n | uniq -c

# There were variants on all chromosomes dropped due to HWE deviations
# from the expected.


#~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
STEP 5.  Remove individuals: missing SNP data and Heterozygosity
#~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~

# 5.1
# Individuals missing SNPs.

# 5.1.1
# Remove individuals missing data on more than 5% of the variants.

plink --bfile cc.qc3 --mind 0.05 --make-bed --out cc.qc4

# Q5a. How many people were dropped?

# Compare this to how many we would have dropped if we removed
# individuals before cleaning up the variants (above in Q3a).

# Cleaning up the variants before checking individuals, means we
# removed only 2.5% of this sample instead of 20% of this sample
# if we had done the reverse.

# 5.1.2
# Plot the comparison in R
# First obtain missingingness information from the data file created
# at the end of STEP 4 when we finished cleaning variants.

plink --bfile cc.qc3 --missing --out miss2

# 5.2
```

```
# Heterozygosity.

# 5.2.1
# 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.qc4 --het small-sample --out het

# 5.2.2
# 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

# 5.2.3
# This awk command will calculate +3SD and -3SD
# (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

# 5.2.4
# Save individuals with heterozygosity rates >3SD into a file to then
drop them.

awk '$5<=0.299429 || $5>= 0.326129' het.txt> het.drop

# Q5b. How many people were >3SD from the mean heterozygosity rate?

# 5.2.5
# Remove them

plink --bfile cc.qc4 --remove het.drop --make-bed --out cc.qc5


#~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
STEP 6.  Prune SNPs
#~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~

# Remember that we postponed removing sex errors until we had
# clean and independent SNPs.

# First prune SNPs. This creates a file (indep.prune.in)
# of independent SNPs following the criteria for
# linkage disequilibrium r2 <0.2 within windows of 1000 kb:

plink --bfile cc.qc5 --indep-pairwise 1000 5 0.2 --out indep


#~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
STEP 7.  Sex check #2
#~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~

# 7.1
```

```
# Check for sex errors using these independent SNPs

plink --bfile cc.qc5 --extract indep.prune.in --check-sex --out sex2

# 7.2
# Copy them into a file to remove from the data set.

grep PROBLEM sex2.sexcheck > sex.drop

# Q7a. How many are there?

# Q7b. Are the others who were initially flagged as mismatches
# still in the data set?

# NOTE: This will code extract all rows that include 'PROBLEM'
# from all files that end in '.sexcheck'

grep PROBLEM *.sexcheck

# Q7c. At what step in QC were the other flagged individuals removed?
# Hint: grep the original IID from the various *fam files
# and you will be able to see at what step they are no longer included.

# 7.3
# Remove the remaining individual(s) with a mismatch on sex.

plink --bfile cc.qc5 --remove sex.drop --make-bed --out cc.qc6

# Q7d. Did this remove the remaining het haploid genotypes?

# 7.4
# 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.qc6 --split-x b37 no-fail --make-bed --out cc.qc7

# 7.5
# 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.

plink --bfile cc.qc7 --set-hh-missing --make-bed --out cc.clean

# Notice that 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 8.  Check Relatedness
#~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
```

```
# 8.1
# Calculate identity-by-descent (IBD) on the pruned SNPs on the
# autosomes. Without actual pedigree data this estimated from
# identity-by-state.

plink --bfile cc.clean --chr 1-22 --extract indep.prune.in --genome --out
ibd

# Look at the output

less ibd.genome

# This calculated the relationships between every pairing of
# individuals in the sample.
# Z0 Z1 Z2 = the pair of IDs share 0, 1 or 2 alleles by descent.
# Ideally 1,0,0 = unrelated; 0,1,0 = parent-child; .25,.5,.25 = siblings
# pihat = proportion IBD = P(IBD=2)+0.5*P(IBD=1)

# 8.2
# Obtain a list of individuals who are related with pihat ≥ .2

plink --bfile cc.clean --extract indep.prune.in --genome --min 0.2 --out
pihat

# Check them in the file pihat.genome
# In this case there are none.


#~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
STEP 9.  Check Ancestry
#~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~

# Use Multidimensional Scaling (MDS) Components to quantify for ancestry.
# MDS clusters individuals together based on how similar they are.
# We will combine our cleaned data with HapMap3 data (MH3),
# then plot the data along MDS components 1 and 2 to see how our
# data 'maps' onto HapMap3 data.

# 9.1
# First drop ambiguous alleles from our cleaned data set and
# Create a file with a list of our variants (rs#).

# NOTE on 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 ambiguous alleles.
# In the 5th and 6th column of the bim file, ambiguous variants 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 ambiguous alleles before merging our data with HapMap3.
# This bit of code selects the possible 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 HM3 data on our SNPs

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

# Obtain a list of HM3 snps
awk '{print $2}' 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

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

# 9.7
# Run the MDS for 4 components

plink --bfile cc.flipped.merged --cluster --mind .05 --mds-plot 4 --
extract cc.clean.snplist --out cc.mds

# 9.8
# Plot the mds in R
```

ANSWERS
Q1a. How is the trait coded? 1 and 2

Q1b. Which build are these data mapped to? build 37 also called hg19.

Q1c. How many SNPs and people are there? 193 individuals: 107 males 86 females. 484128 variants

Q2a. How many mismatches were flagged? 3

Q3a. How many individuals will we drop if we keep a Sample Call Rate of >=95 (i.e. remove those with >5% missing SNPs)? 39 individuals

Q4a. How many variants were removed? 130901 variants removed.

Q4b. How many variants were removed due to MAF? 49709 variants.

Q4c. How many variants were dropped due to HWE? 626 variants.

Q5a. How many people were dropped? 5 individuals.

Q5b. How many people were >3SD from the mean heterozygosity rate? 3 individuals.

Q7a. How many are there? 1

Q7b. Were the others who were initially flagged as still in the data set? No

Q7c. At what step in QC were the other flagged individuals removed? Those flagged with heterozygosity proportions of approximately .5 were cleaned out when the individuals with too much missing data were dropped.

Q7d. Did this remove the remaining het haploid genotypes? No, there are still 3.

Q7e. Why did the number of variants or individuals not change? The errors come from specific variants for certain people. Only those errors are set to missing.