**BIOMI 609 Computational Genomics and Bioinformatics**

**San Diego State University**

**Dr. Arun Sethuraman**

**Lab 5 Genome Wide Association Studies**

This week, we will learn how to perform GWAS and interpret results from them. The dataset we will start with is one that describes congenital deafness in three breeds of dogs from Hayward et al., 2020 (<https://doi.org/10.1371/journal.pone.0232900>). Briefly, they wish to understand the genomic basis of deafness in Dalmatians, Australian Cattle dogs, and English Setters using genome-wide SNP data, combined with deafness phenotypes. I have posted this paper on Canvas for your reference. We will be utilizing data from this paper that can be downloaded from Dryad: <https://doi.org/10.5061/dryad.sf7m0cg2n>.

**Exercise 0: Setting up data files on JetStream**

Open your JetStream VM, create a new folder (say called Week13), and download the dataset (as a zip file), thereon cd into that folder via a terminal window and unzip it. You can do this using:

unzip doi\_10.5061\_dryad.sf7m0cg2n\_\_v3.zip

You’ll see that there are 4 files - deafness.bed (binary file containing all the SNP’s - note that the BED format is not readable), deafness.bim (containing information on all the genetic markers), deafness.fam (containing information about individuals), and deafness\_pheno.txt (containing phenotypic information). From their README file, these are specifically:

Binary PLINK files containing genotype data of 503 dogs from three breeds (dalmatian, Australian cattle dog, english setter) run on the semi-custom 220k CanineHD array (Illumina), with positions in CanFam3.1 reference. After quality control filters, 201,020 SNPs remain.

deafness.fam - ID and sex (if known) information for each dog

deafness.bim - chromosome, name, bp location, alleles for each SNP

deafness.bed - genotypes for each dog at each SNP

Phenotype file with dog ID, sex (0 if unknown), breed, location, BAER test hearing phenotype, and sibling-pair GWAS design phenotype - deafness\_pheno.txt

We have already installed plink2, the software that we will be utilizing to do our GWAS - but please double check that this is in your PATH. To check:

echo $PATH

(This should show the path to plink2 - if not, please check where you’ve installed it and add it to your path accordingly).

Also, go ahead and download the population files (acd.txt, dalmatian.txt, setter.txt) files from Canvas to your folder. These just contain the individual ID’s of each breed.

Please also go ahead and download a “modified” version of the deafness\_pheno.txt to replace the old one (I’ve uploaded this to Canvas) for us to be able to run plink2 on these data. The only changes that I’ve made are:

1) Changed “dogID” to “#IID” - this is required by plink2

2) Changed the categorical variables (deafness phenotypes), such that “hearing” is coded as 0, “bilaterally\_deaf” is coded as 2, and “unilaterally\_deaf” is coded as 1 (i.e. additive phenotype).

**Exercise 1: Quality Control of the data**

Best practices for running any GWAS are detailed in this superb manuscript: Marees et al., 2018 (DOI: [10.1002/mpr.1608](https://dx.doi.org/10.1002%2Fmpr.1608)). Here’s a table from this manuscript that describes all the steps of QC that we ideally should be doing before running our GWAS.

We are going to visualize population structure using a PCA, and then do several of these steps at once, prior to doing a GWAS.

To do a PCA on the data, to visualize structure:

plink2 --bfile deafness --pca --aec --chr-set 40 --out deafness

This will print out two files: deafness.eigenval and deafness.eigenvec, which we can then visualize using R. So go to R (just type R and hit enter).

Thereon, read the eigenvec file, followed by the population designation files.

eigenvec<-read.table("deafness.eigenvec")

dalmatian<-read.table("dalmatian.txt",header=FALSE)

setter<-read.table("setter.txt",header=FALSE)

acd<-read.table("acd.txt",header=FALSE)

d<-which(eigenvec$V1%in%dalmatian$V1)

a<-which(eigenvec$V1%in%acd$V1)

s<-which(eigenvec$V1%in%setter$V1)

Now let’s plot this!

plot(eigenvec$V3,eigenvec$V4,xlab="PC1",ylab="PC2",col="white")

points(eigenvec$V3[d],eigenvec$V4[d],col="red")

points(eigenvec$V3[s],eigenvec$V4[s],col="blue")

points(eigenvec$V3[a],eigenvec$V4[a],col="green")

Chart, scatter chart

Description automatically generated

Here you clearly see that the populations are clustering within their own breeds (blue = setters, green = Australian Cattle Dogs, red = dalmatians). Thankfully there are no significant admixed individuals (see the Marees et al., paper for an example of individuals that we should ideally nix from further analyses to avoid spurious associations due to structure). So we can now proceed to our next set of filtering:

Let’s filter for --maf cutoff of 0.05, --geno (missing data) of > 20%, --hwe (loci out of HWE) at a p-value cutoff of 0.05, and --king-cutoff of 0.2 (removing individuals that are very close relatives only - i.e. relatedness > 0.2).

plink2 --bfile deafness --maf 0.05 --geno 0.02 --hwe 0.05 --out deafness\_filtered --make-bpgen --aec --chr-set 40 --king-cutoff 0.2

Note that plink2 will now write a separate set of filtered .fam, .bim, and .pgen files called “deafness\_filtered”.

**Exercise 2: Running a GWAS**

Now running a GWAS is as simple as using the command below - here --bfile specifies the names of the .bim, .bed, .fam files to be used, the --pheno tag specifies the phenotype file to be used, --pheno-col-nums is the column number that contains the phenotype that we are interested in analyzing, --aec allows extra chromosomes (other than those in humans), --chr-set sets the total number of chromosomes present in the file, --out is the name of the output file to be produced, and --no-fid states that there is no family ID present in the phenotype file. So go ahead and run this:

plink2 --bfile deafness\_filtered --pheno deafness\_pheno.txt --pheno-col-nums 5 --linear --aec --chr-set 40 --out deafness\_gwas --no-fid --pgen deafness\_filtered.pgen

Now let’s take a look at the output file: deafness\_gwas.BAER\_test\_phenotype.glm.logistic

In order for us to use the qqman package to plot these, we’ll have to go ahead and remove the # at the beginning of this file. So just do this using:

sed -i 's/#//g' deafness\_gwas.BAER\_test\_phenotype.glm.logistic

Exercise 2: Making a Manhattan Plot

Now go into R again.

library(qqman)

deafness<-read.table("deafness\_gwas.BAER\_test\_phenotype.glm.logistic",header=TRUE)

deafness\_nomissing<-na.omit(deafness) #remove missing entries

manhattan(deafness\_nomissing,chr="CHROM",bp="POS",p="P",snp="ID",logp=TRUE,ylab="log10 p values",xlab="SNPs",chrlabs=c(1:40),col=c("blue4","orange3"))

Chart, scatter chart

Description automatically generated

We can now summarize the top outlier loci (interesting how this doesn’t line up with the findings of the original study - think why!).

Now try running the same analyses with the original data (without filtering), and other combinations! Voila!