GINGER Plink Exercises (UCT 2022 Training)
Kumar Veerapen, Ph.D. and Carla Marques-Luna, Ph.D.

Adapted from Elizabeth Atkinson, Ph.D.

Amended from two tutorials that the PLINK author Shaun Purcell wrote, see http://pngu.mgh.harvard.edu/~purcell/plink/tutorial.shtml and 'Teaching materials and example dataset' at http://pngu.mgh.harvard.edu/~purcell/plink/res.shtml

In this tutorial, we will use PLINK to analyze some real large-scale SNP data, to give a demonstration of what the program can do (e.g. data management, summary statistics and basic association analysis). What we do today might not be particularly realistic or accurate, but we hope that it gives an idea of what PLINK is capable of!

EXAMPLE DATASETS AND SOFTWARE

Ensure that your docker image is running
`us.gcr.io/gingeriimak/ginger-uct:0.1`

We're using the plink files that Alicia Martin and Zan Koenig created found here and can be copied into your working directory as follows

```
gsutil cp gs://neurogap_phenos_genos/NeuroGAP_pilot_clean_grch38.*
.
```

CARDINAL RULES & CAVEATS

When using *PLINK* there are a few key points to remember.

- Always consult the LOG file (console output)
- *PLINK* has no memory
  - each run loads data anew, previous filters lost
- Exact syntax and spelling is **very important**
- Not every option can be combined with every other option
  - For example, basic haplotype tests cannot take covariates
  - *PLINK* doesn't always warn you
  - LOG file often shows what has happened (or not)
- Consult the web documentation (http://pngu.mgh.harvard.edu/purcell/plink/)

GETTING STARTED

PLINK is a command line program, so we need to operate in a command line window. All commands involve typing `plink` at the command prompt (e.g. DOS window or Unix terminal) followed by a number of options (all starting with `--option`) to specify the data files / methods to be used. All results are written to files with various extensions.

<u>In italics are materials if you do not have docker running</u>:
Putting your files in the same directory as the "plink.exe" file will let you do all analysis from this directory. Navigate to the plink directory, and to check you are in the correct folder, type

```
plink
```

and press enter, which should start PLINK and generate some output describing the program. If you get an error message, you are in the wrong directory. Alternatively, make a new directory to store all the files you generate throughout this tutorial using the Linux skills you learned yesterday and call plink from where it lives.
Note: if you don't have plink on the path, you might need to use the full path to your plink program to call the program throughout this tutorial.

For most options, PLINK needs three plink binary files: .bed, .bim, .fam files

It is easiest if these files have the same name (e.g. sheep.ped and sheep.map).

There are lots of options to change the format of these files (for example, you can provide genotypes as "AG" instead of "A G") - see the information on the PLINK website.

PLINK is designed for humans - so unless you tell it otherwise, it will assume your genome has 23 chromosomes, with chr23 being the X chromosome! Use the option --dog for up to 39 chromosomes.

NeuroGAP Pilot Genotyping

In this tutorial, we will be using example data that is the NeuroGAP pilot genotyping datasets, which consist of the first 192 samples from each NeuroGAP site genotyped on the Illumina Global Screening Array. (Due to sample QC, not all 960 individuals genotyped are included here.)

The following is the basis of all of your commands in this tutorial:

```
plink --bfile NeuroGAP_pilot_clean_grch38
```
.

plink will typically output files to disk. If you don't specify an output name, it will use the name 'plink'. You can change the basename of your output with the --out option. So for example, to send the log file to neuro1.log, you could do the following:
plink --bfile NeuroGAP_pilot_clean_grch38 <insert comman> --out neuro1

Other notes:

- As we explore other plink options, you will see that plink uses different suffixes (e.g., *frq*, *assoc*), all with the same base name, given by the --out option.
- Sometimes you may use the same input file with slightly different options and not want to over-write pre-existing files. The best approach is to use sensible names.

All output files that PLINK generates have the same format: root.extension where root is, by default, "plink" but can be changed with the --out option, and the extension will depend on the type of output file it is.

Now let's do some simple statistical analysis! Run the command:

```
  plink --bfile NeuroGAP_pilot_clean_grch38 --freq --out neuro1 -
-allow-extra-chr
```

```
PLINK v2.00a3LM 64-bit Intel (28 Mar 2022)     www.cog-
genomics.org/plink/2.0/
(C) 2005-2022 Shaun Purcell, Christopher Chang   GNU General
Public License v3
Logging to neuro1.log.
Options in effect:
  --allow-extra-chr
  --bfile NeuroGAP_pilot_clean_grch38
  --freq
  --out neuro1

Start time: Mon Apr  4 21:54:29 2022
3922 MiB RAM detected; reserving 1961 MiB for main workspace.
Using up to 2 compute threads.
900 samples (441 females, 459 males; 900 founders) loaded from
NeuroGAP_pilot_clean_grch38.fam.
342962 variants loaded from NeuroGAP_pilot_clean_grch38.bim.
Note: No phenotype data present.
Calculating allele frequencies... done.
--freq: Allele frequencies (founders only) written to neuro1.afreq
.
End time: Mon Apr  4 21:54:29 2022
```

Look at the output file and make sure you understand it. From the plink documentation online, what does the –freq command do? What is shown in your output?

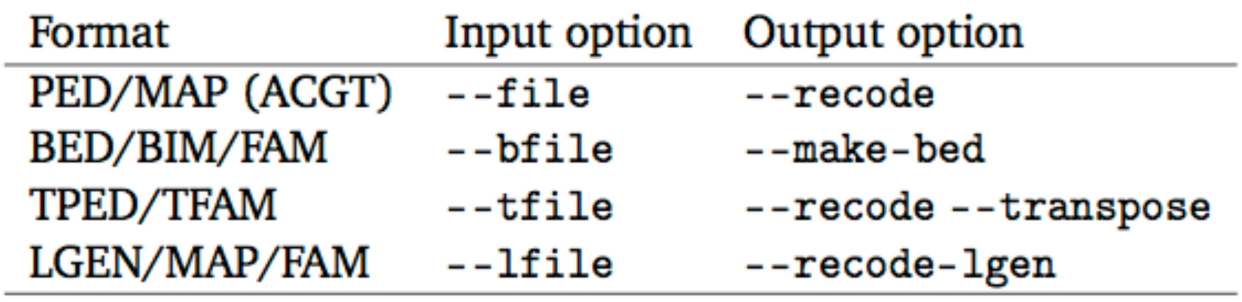
The information contained here can be summarized as follows:

- A banner showing copyright information and the version number -- the web-based version check shows that this is an up-to-date version of PLINK and displays a message that v0.99l is a pre-release testing version.
- A message indicating that the log file will be saved in neuro1.log. The name of the output file can be changed with the --out option as described above
- A list of the command options specified.
- By keeping track of log files, and naming each analysis with its own --out name, it makes it easier to keep track of when and how the different output files were generated.
- What do you think are the other lines for?
- Finally, a line is given that indicates when this analysis finished. You can see that it took 8 seconds (on my machine at least) to read in the file and apply the filters.

If other analyses had been requested, then the other output files generated would have been indicated in the log file.

Recoding

Data can also be transformed into other formats. Some recoding options are shown in the table below.



Format	Input option	Output option
PED/MAP (ACGT)	`--file`	`--recode`
BED/BIM/FAM	`--bfile`	`--make-bed`
TPED/TFAM	`--tfile`	`--recode --transpose`
LGEN/MAP/FAM	`--lfile`	`--recode-lgen`

Note that for the PED format, alleles can be encoded as ACGT or 1234. The --alleleACGT and --allele1234 options can be used to do conversion, but you have to use the --recode or --make-bed output flags, too.

Summary statistics: missing rates

Next, we will generate some simple summary statistics on rates of missing data in the file, using the -- missing option:

```
plink --bfile --bfile NeuroGAP_pilot_clean_grch38 --missing --out miss_stat --allow-extra-chr
```

which should generate the following output:

```
...
900 samples (441 females, 459 males; 900 founders) loaded from
NeuroGAP_pilot_clean_grch38.fam.
342962 variants loaded from NeuroGAP_pilot_clean_grch38.bim.
Note: No phenotype data present.
```

```
Calculating sample missingness rates... done.
Calculating allele frequencies... done.…
```

The per individual and per SNP (after excluding individuals on the basis of low genotyping) rates are then output to the files miss_stat.smiss and miss_stat.vmiss respectively.

These output files are standard, plain text files that can be viewed in any text editor, pager, spreadsheet or statistics package (albeit one that can handle large files). Taking a look at the filemiss_stat.vmiss, for example using the 'more' command, which is present on most systems:

```
more miss_stat.vmiss
```

we see:

```
CHR         SNP  N_MISS    F_MISS
  1   rs6681049       0         0
  1   rs4074137       0         0
  1   rs7540009       0         0
  1   rs1891905       0         0
  1   rs9729550       0         0
  1   rs3813196       0         0
  1   rs6704013       2 0.0224719
  1    rs307347      12  0.134831
  1   rs9439440       2 0.0224719
...
```

THINt from more, type 'q' to quit

That is, for each SNP, we see the number of missing individuals (N_MISS) and the proportion of individuals missing (F_MISS). Similarly:

```
more miss_stat.smiss
```

we see:

```
FID              IID  MISS_PHENO  N_MISS    F_MISS
HCB181             1           N     671 0.0080326
                                                 6
HCB182             1           N    1156 0.0138387
HCB183             1           N     498 0.0059616
                                                 4
HCB184             1           N     412 0.0049321
                                                 2
HCB185             1           N     329 0.0039385
                                                 2
HCB186             1           N    1233 0.0147605
HCB187             1           N     258 0.0030885
                                                 6
...
```

The final column is the actual genotyping rate for that individual -- we see the genotyping rate is very high here.

If you are using a spreadsheet package that can only display a limited number of rows (some popular packages can handle just over 65,000 rows) then it might be desirable to ask PLINK to analyze the data by chromosome, using the --chr option. For example, to perform the above analysis for just chromosome 1:

```
plink --bfile NeuroGAP_pilot_clean_grch38 --chr 1 --out res1 –missing --allow-extra-chr
```

To reiterate, the --missing option makes plink compute how many of the calls are missing. In the output file with the .smiss suffix, for each individual we are told what how many SNP calls are missing and what the missing rate is. In the file with the .vmiss file, we see listed for each SNP, the number of individuals for which the call for that SNP is missing.

Exercise: Find the ten SNPs in the NeuroGAP pilot file with the highest missing rate.

Hint: after running plink, use the Linux sort, head and cut commands or similar
[one way: cat miss_stat.lmiss | sort -k 5 -n -r | head -n 10 |awk '{ print $2}']

NOTE: The --test-missing option makes plink see whether the missing rate in cases and controls are significantly different. A $\chi 2$ test is used. This a very important test to use in principle. In practice, we are often very strict about the missing rates and filter out any SNPs that have a significant missing rate – hence relatively few SNPs will fail this --test-missing test since we've already excluded any with significant missing rate. But one should use this as a sanity test.

Summary statistics: allele frequencies

Next we'll perform a similar analysis, except requesting allele frequencies instead of genotyping rates. The following command generates a file called freq_stat.frq which contains the minor allele frequency and allele codes for each SNP.

```
plink --bfile NeuroGAP_pilot_clean_grch38 --freq --out freq_stat --allow-extra-chr
```

It is also possible to perform this frequency analysis (and the missingness analysis) stratified by a categorical, cluster variable, that you can create from GTS how to do it with plink 😊

Extracting only entries for particular SNPs

If you were just interested in a specific SNP, and wanted to know what the frequency was in the two populations, you can use the --snp option to select this SNP:

```
    plink --bfile NeuroGAP_pilot_clean_grch38 --snp 1:9348900:A:C --freq --out snp1_frq_stat --
allow-extra-chr
```

would generate a file snp1_frq_stat.afreq containing only the frequencies for this single SNP. You can also specify a range of SNPs by using the options --from-bp and --to-bp with genomic basepair coordinates to extract out just a smaller region of interest of a chromosome.

Try it! Write the window you are extracting here. Don't forget to specify which chromosome you're looking at with the --chr flag. How many SNPs were included in your region?

These commands are only useful for a few SNPs. To extract many SNPs,

1) put the SNP IDs you want to extract into a file (one column, no header, just SNP IDs). Name it whatever you'd like, here we are using snplist.txt
2) and use the --extract sequence. Try this:

```
    plink --bfile NeuroGAP_pilot_clean_grch38 --extract snplist.txt --make-bed --out extract –
allow-extra-chr
```

Note how you must use --extract in conjunction with a recode or make-bed command to make a new file.

If you wanted to remove these SNPs rather than keep them, the --exclude option does the inverse of this.

Extracting named individuals

Similar to the use with subsetting sites, the --keep and --remove flags take file names as arguments. The corresponding files should contain the individuals each on a line with family and individual ID. For example, in the file example that you can create which would look like the following vip.txt we have:

```
HCB193 1
HCB194 1
HCB195 1
HCB196 1
```

```
HCB197 1
HCB198 1
```

To extract just these individuals, we can say:

```
plink --bfile NeuroGAP_pilot_clean_grch38 --keep vip.txt --make-bed  --out subset
```

You can combine options too! For example you could also use the --extract flag to keep specific sites in addition to specific individuals. This combo command would extract out the details of a few individuals and a few of their SNPs and stores the extracts in a new file.

Explore linkage disequilibrium

plink has various routines to deal with linkage disequilibrium   The most basic is the one to compute LD between two named SNPs. For example :

```
plink --bfile NeuroGAP_pilot_clean_grch38 --ld 1:4229839:T:C 1:38694218:C:T --allow-extra-chr
```

What does this output mean??

The --indep-pairwise option allows us to *prune* SNPs that are in LD. What this does is scan through the genome and try, in each window, to select one SNP that represents the others in the window. It takes three arguments — the length of the sliding window, how much the window is shifted at each step, and the r2 value which is used.   For example:

```
plink --bfile NeuroGAP_pilot_clean_grch38 --indep-pairwise 50 10 0.1 --out check --allow-extra-chr
```

The file check.prune.in is the list of SNPs that should be used as representative of the data

The file check.prune.out is the list of SNPs that could be filtered out.

The --extract or --exclude options can be then used to appropriately exclude redundant SNPs.

Selecting based on criteria

Earlier, we saw we could extract or remove based on SNP ID or individual ID. Now we look at removing based on various criteria.

–mind excludes individuals with missing genotype data above the given rate. (This is an example – apply your mind to your Hapmap1 data and choose appropriate file names and parameters)

```
plink --bfile NeuroGAP_pilot_clean_grch38 --mind 0.04 --make-bed --out
data-i96 --allow-extra-chr
```

This creates a new set of binary ped files *data-i96* that contain all individuals from the *data* file who have at least a 96% call rate. The IDs of the individuals removed are put in the *.irem* file.

The --geno option removes SNPs that have less than the specified call rate

```
 plink --bfile NeuroGAP_pilot_clean_grch38 --geno 0.04 --make-bed --out
data-s96 --allow-extra-chr
```

Exercise: pick a genotyping rate and see how many SNPs pass this threshold in your dataset. Write your filtering threshold and the number of individuals and SNPs which pass here:

The --maf option can be used to remove SNPs with a MAF below a certain rate.
Exercise : What if you just want sites with a MAF rate of at least 0.4? How many SNPs pass this cutoff?

You can also use the --hwe option to remove SNPs which fail Hardy-Weinberg.

PART 2: Basic association analysis

Let's now perform a basic association analysis on the disease trait for all single SNPs. All the earlier preparatory commands are crucial to ensure good quality data.

The basic command is (you almost never run such a vanilla association without any covariates)

```
plink --bfile NeuroGAP_pilot_clean_grch38 --glm allow-no-covars --out as1 --allow-extra-chr
```

which generates an output file as1.assoc which contains the following fields

```
CHR        SNP A1    F_A    F_U  A2     CHISQ         P     OR
  1   rs668104 1 0.1591 0.2667   2     3.067   0.07991 0.5203
             9
  1   rs407413 1 0.0795 0.0777   2  0.001919    0.9651  1.025
             7        5      8
  1   rs189190 1 0.4091    0.4   2   0.01527    0.9017  1.038
             5
  1   rs972955 1 0.1705 0.0888   2     2.631    0.1048  2.106
             0               9
  1   rs381319 1 0.0340 0.0222   2    0.2296    0.6318  1.553
             6        9      2
  1   rs120445 1    0.5 0.4889   2   0.02198    0.8822  1.045
            97
  1   rs109071 1 0.3068 0.2667   2    0.3509    0.5536  1.217
            85
  1   rs112606 1 0.2326    0.2   2    0.2754    0.5998  1.212
            16
  1   rs745910 1 0.1395 0.1932   2    0.9013    0.3424 0.6773
...
```

where each row is a single SNP association result. The fields are:

- Chromosome
- SNP identifier
- Code for allele 1 (the minor, rare allele based on the entire sample frequencies)
- The frequency of this variant in cases
- The frequency of this variant in controls
- Code for the other allele
- The chi-squared statistic for this test (1 df)
- The asymptotic significance value for this test
- The odds ratio for this test

If a test is not defined (for example, if the variant is monomorphic but was not excluded by the filters) then values of NA for *not applicable* will be given (as these are read by the package R to indicate missing data, which is convenient if using R to analyse the set of results).

HIN In a Unix/Linux environment, you can use the available command line tools to sort the list of association statistics and print out the top ten, for example:

```
sort --key=7 -nr as1.assoc | head
```

would give:

```
rs9585021   1   0.625 0.28412       20.625.586e-06     4.2

rs2222162   1   0.284 0.62222       20.515.918e-06  0.2409
                   1
```

More on association analysis with Carla 😊 <3

Extracting a SNP of interest

Finally, given you've identified a SNP, set of SNPs or region of interest, you might want to extract those SNPs as a separate, smaller, more manageable file. In particular, for other applications to analyse the data, you will need to convert from the binary PED file format to a standard PED format. This is done using the --recode options. There are a few forms of this option: we will use the --recode12 that codes the genotypes in a manner that is convenient for subsequent analysis. To extract only this single SNP, use:

```
plink --bfile NeuroGAP_pilot_clean_grch38 --snp 1:9348900:A:C --make-bed --out rec_snp1 -
-allow-extra-chr
```

This particular recode feature codes genotypes as 1/2 alleles, and outputs new .ped and .map files with this SNP.