Exercises

A- **Take a look at SNP and get acquainted with the use of a few Unix-Linux tools**:

Go to mice_cours/data

```
$cd mice_cours/data
```

Look the genotype file

```
$less –S mice_genotypes.txt
```

This file is prepared to be used in GS3 and blupf90, and it is in fixed format. This is inconvenient because many programs (plink?) admit SNPs separated by spaces. However, some Fortran compilers (e.g. Intel) don't admit free format beyond a large number of columns. So it is better to compact the genotype. This also saves lots of space.

The format is pretty easy to handle using Awk, Fortran, and scripts (and possibly Perl, Python, etc.).

Alleles are codified as gene content {0,1,2} for genotypes {AA, Aa (or aA), aa}. What allele is being counted ("a" in this case) is arbitrary. We do codify missing genotypes as 5. This format comes from Paul VanRaden, slightly modified. I'll call it "UGA format".

It's important to understand what fixed format means. It means that id's go from position *i* to *j* and genotypes from *k* to *l* always. The typical error is not to take into account that id's have variable lengths, e.g., this file would not be correctly read:

```
45 1111212…
346 1121111…
347 2022222…
348 1111111…
349 2022222…
1350 1111212…
```

Whereas this is a correct format:

```
  45 1111212…
 346 1121111…
 347 2022222…
 348 1111111…
 349 2022222…
1350 1111212…
```

How to detect if our file is correctly formatted or not? I usually check that all lines have the same length:

```
$awk '{print length($0)}' mice_genotypes.txt | sort -u
```

You should find a single figure as answer (i.e., all lines have same length).
So, how many genotyped animals?

```
$wc –l mice_genotypes.txt
```

How many SNP genotyped?

```
$awk '{print length($2)}' mice_genotypes.txt
```

Everyone has the same number of loci in the data file?

```
$awk '{print length($2)}' mice_genotypes.txt | sort -u
```

Let's extract the first 200 individuals *while keeping the format.* This is a one-liner, beware of simple and double quotes !!

```
$awk ' NR <= 200 {printf("%10s %1s%" length($2) "s\n", $1, " ",  $2) }' mice_genotypes.txt
> temp
```

Note that the format is defined by `"%10s%1s%" length($2) "s\n"` which means "10 positions, 1 position (for the space in " " later), as many positions as SNPs we have for each individual (in `%" length($2) "s`), and the line return in `\n`. Otherwise awk would change our beautifully crafted format ☺ .

Extracting one SNP in whatever column we like (e.g., column 16). This is very useful for GWAS scripting (we'll come back to this later):

```
$awk ' {printf("%10s%1s%1s\n", $1, " ",  substr($2,16,1) }' mice_genotypes.txt > temp
```

B-
Impose a a MAF (minor allele frequency).

Some people like imposing MAF's. This is arguable because SNPs that are monomorphic do not add anything, but they don't cause trouble either. There is nevertheless a Fortran program in

```
$cd ../progs
```

You should take a look at the program: (choose your own editor)

```
{vim nano  joe nedit gedit emacs} filter_maf.f90
```

The program reads everything in memory, computes allelic frequencies and writes out loci with MAF > the desired one. The program considers missing genotypes properly.
Now, let's impose MAF >= 3%.

```
$cd ../data
$../progs/filter_maf
$ ../progs/filter_maf
 genofile?
```

Type:

```
mice_genotypes.txt
 minimum MAF?
```

Type:

```
0.03
 Column position in file for the first marker:           12
 Format to read SNP file:
 (i10,1x,10946i1)

 Number of SNPs :       10946
 nanim=         1884 nsnp=        10946
        1000
       10000
 missing genotypes  0.000000000000000E+000
 average freq=   0.512276949805317
 sdfreq=   0.277650417982918
 maxfreq=    1.00000000000000
 minfreq=   0.000000000000000E+000
 average MAF=   0.261874811611373
 sdMAF=   0.143306918037798
 maxMAF=   0.499734607218684
 minMAF=   0.000000000000000E+000
 number of SNP over MAF>  3.000000000000000E-002  is
10365
       10000
  -- After cleaning out by MAF --
 average freq=   0.483726544282432
 sdfreq=   0.278767144612452
```

```
maxfreq=   0.969214437367304
minfreq=   3.025477707006369E-002
average MAF=   0.261125039423411
sdMAF=   0.144619153247089
maxMAF=   0.499734607218684
minMAF=   3.025477707006369E-002
```

Three new files have been created:
`mice_genotypes.txt_MAF03`
with genotypes with MAF > 3% and
`mice_genotypes.txt_MAF03freq`
with their frequencies (from this file it is easy to compute quantities as $2\sum p_i q_i$, $\bar{p}$ and $Var(p)$), and
`mice_genotypes.txt_freq`
with the “old” frequencies and the correspondence between old and new.

C

Prediction using SNPs in R (can be done in Windows)

Go to `mice_data_R`

There is an R program that estimates SNP effects using this mice data. It uses BLUP_SNP, and the variance components are those in Legarra et al. 2008 (Genetics):

`blupgenomic_crossval.r`

There are three data sets, including 1095, 10946, and 5473 SNPs.

In the same directory, open R. Copy and paste commands until line

```
# ------------------------------------------------
# END OF FIRST PART
# ------------------------------------------------
```

You will see, first, the computation time of solving BLUP_SNP (it takes 17 s in my "old" (>3 years) laptop); and second, a result (`print(cor(y,yhat)):` an empirical correlation between the EBV and raw phenotypes. More or less SNPs can be included setting the variable "set" to different values, in

```
set=1  #10946
   #2  #5473
   #3 1095
```

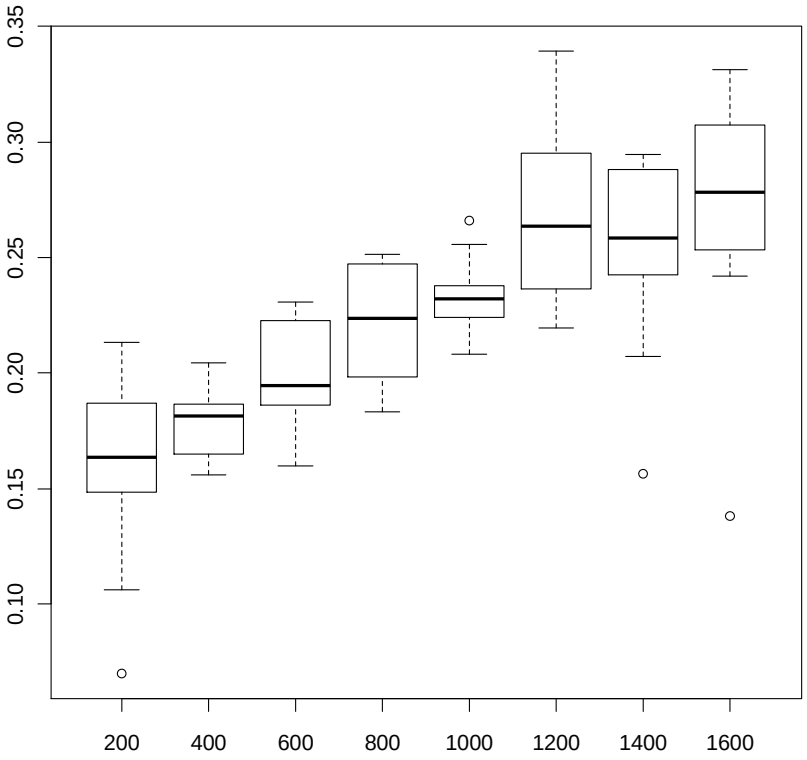
(don't forget to re-read the whole thing again, i.e., submit the lines from the top of the program).

There are two methods, PCG and GSRU; in my computer PCG takes 17 s … find out how long does it take for GSRU (you can change the convergence criteria as well).

However, this correlation is way too high because we are predicting "training" data. What we're interested in is in prediction of unseen phenotypes (or progeny performances). So, the next piece of program does crossvalidation, splitting animals in training and validation, with increasing training sizes, and 10 replicates for each size of the training data set (R is wonderful for doing this kind of things).
So run the script until

```
# ------------------------------------------------
# END OF SECOND PART
# ------------------------------------------------
```

A Graphics "correlation between size of training population" is generated:



You can also do `plot(size, pp)` Note that more training implies more accuracy, but it seems to be asymptotic.

On the other hand, look at the dispersion of correlations for a given training size. It is fairly disperse; this means that a punctual estimate of this correlation can be rather inaccurate, in special if we have too few individuals in either the training or the validation data set (e.g., at the extremes of the graph). This shows nicely that one should not try to draw too many conclusions from cross-validation analysis.

What you can do next is to play with the amount of SNPs and/or variances. Also there are lots of diagnostics that can be done.