# The data set

**Description of files:**

**data3.txt:**
1: animal ID
2: generation
3: sex
4: phenotype
5: true breeding value (TBV)

**ped3.txt:**
1: animal ID
2: sire ID
3: dam ID

**snp3.2k:**
1: animal ID
2: SNP genotype

**mrkmap.txt:**
1: SNP ID
2: Chromosome
3: position

# Block 1

1. Files are available on the website. Use curl to download it to your Linux or Mac device:

```
curl http://nce.ads.uga.edu/wiki/lib/exe/fetch.php?media=lab3_UF.zip -o lab3.zip
```

1. Run renumf90 program using renum.par parameter file to renumber the data.

# Block 1

2. Single-step GWAS (ssGWAS): SNP effects computed based on GEBV are used to compute SNP weights or variance explained by SNP. The default way to calculate SNP weight (w) in postGSf90 is:

$$w_i = 2p_i(1-p_i)a_i^2$$

where p is the allele frequency and $a$ is SNP effect. A new method that has better convergence properties has been recently implemented in **postGSf90**. This method is called nonlinearA and is described in VanRaden (2008) as:

$$w_i = \mathrm{CT}^{\frac{|\widehat{a_i}|}{sd(\hat{a})}-2}$$

where CT is a constant set to 1.125, and $\frac{|\widehat{a_i}|}{sd(\hat{a})}$ is capped to 5 by default. To use this method, the following option should be used in postGSf90:

```
OPTION which_weight nonlinearA
```

Run postGSf90 including an option to calculate variance based on a window of 20 SNPs and an option to generate Manhattan plots. Use the default linear weight and the nonlinearA weight. Check the output files and compare results.

# Block 1

Hint 1: Although variance explained by SNP (SNP weight) is useful in the context of breeding and genetics, p-values can be computed to declare significance of SNP. This computation requires the inverse of the LHS of the MME (Aguilar et al., 2019 - https://doi.org/10.1186/s12711-019-0469-3). To obtain p-values for SNP, both blupf90 and postGSf90 should include the following option:

```
OPTION snp_p_value
```

Hint 2: `postGSf90` prints Manhattan plots on the screen and also creates files for printing in R (Sft1e2.R, Vft1e2.R, Pft1e2.R) and in Gnuplot (Sft1e2.gnuplot, Vft1e2.gnuplot and Pft1e2.gnuplo).

# Block 1

Hint 3: Check all the options related to GWAS here:
http://nce.ads.uga.edu/wiki/doku.php?id=readme.pregsf90

Hint 4: Although this exercise was based on ssGBLUP, you can use the tricks from yesterday to switch to GBLUP.

1. Files are available on the website. Use curl to download it to your Linux or Mac device:

```
curl http://nce.ads.uga.edu/wiki/lib/exe/fetch.php?media=lab3_UF.zip -o lab3.zip
```

1. Run renumf90 program using renum.par parameter file to renumber the data.

renum.par

```
DATAFILE
data3.txt
TRAITS
 4
FIELDS_PASSED TO OUTPUT
2
WEIGHT(S)

RESIDUAL_VARIANCE
0.60
EFFECT
3 cross alpha
EFFECT
1  cross alpha
RANDOM
animal
FILE
ped3.txt
FILE_POS
1 2 3 0 0
SNP_FILE
snp3.2k
PED_DEPTH
0
INBREEDING
pedigree
(CO)VARIANCES
0.40
OPTION map_file mrkmap.txt
```

Data provided:

```
data3.txt
mrkmap.txt
ped3.txt
snp3.2k
```

```
# Running renumf90 and saving the log file
renumf90 renum.par | tee renum.log
```

2. Run postGSf90 including an option to calculate variance based on a window of 20 SNPs and an option to generate Manhattan plots. Use the default linear weight and the nonlinearA weight. Check the output files and compare results.

ssgblup.par

```
# BLUPF90 parameter file created by RENUMF90
DATAFILE
 ../renf90.dat
NUMBER_OF_TRAITS
           1
NUMBER_OF_EFFECTS
           2
OBSERVATION(S)
    1
WEIGHT(S)

EFFECTS: POSITIONS_IN_DATAFILE NUMBER_OF_LEVELS
 2          2 cross
 3      12010 cross
RANDOM_RESIDUAL VALUES
  0.60000
 RANDOM_GROUP
      2
 RANDOM_TYPE
 add_an_upginb
 FILE
../renadd02.ped
(CO)VARIANCES
  0.40000
OPTION SNP file ../snp3.2k
OPTION map_file ../mrkmap.txt
OPTION no_quality_control
OPTION saveGInverse
OPTION saveA22Inverse
```

**LINEAR WEIGHTS**

```
# LINEAR WEIGHTS
mkdir linear ; cd linear
# Run blupf90 to save Gi, A22i and obtain solutions
cp ../renf90.par ssgblup.par .  #copying and renaming

echo ssgblup.par | blupf90 | tee blup.log
```

2. Run postGSf90 including an option to calculate variance based on a window of 20 SNPs and an option to generate Manhattan plots. Use the default linear weight and the nonlinearA weight. Check the output files and compare results.

postgs_lin.par

**LINEAR WEIGHTS**

```
# BLUPF90 parameter file created by RENUMF90
DATAFILE
 ../renf90.dat
NUMBER_OF_TRAITS
           1
NUMBER_OF_EFFECTS
           2
OBSERVATION(S)
    1
WEIGHT(S)

EFFECTS: POSITIONS_IN_DATAFILE NUMBER_OF_LEVELS
 2          2 cross
 3      12010 cross
RANDOM_RESIDUAL VALUES
  0.60000
 RANDOM_GROUP
      2
 RANDOM_TYPE
 add_an_upginb
 FILE
../renadd02.ped
(CO)VARIANCES
  0.40000
OPTION SNP_file ../snp3.2k
OPTION map_file ../mrkmap.txt
OPTION no_quality_control
OPTION readGInverse
OPTION readA22Inverse
OPTION Manhattan_plot
OPTION windows_variance 20
```

```
# Run postGSf90
cp ../renf90.par postgs_lin.par
echo "OPTION Manhattan_plot" >> postgs_lin.par
#Variance per window of 20 SNP
echo "OPTION windows_variance 20" >> postgs_lin.par
echo postgs_lin.par | postGSf90 | tee postgs.log

#TWO Manhattan plots will be generated:
-variance by snp
-variance by window

***** TIP *********
#The two graphs will be displayed automatically after postgsf90
#However, they can be displayed at any time using the command:
gnuplot Sftle2.gnuplot
gnuplot Vftle2.gnuplot
#The same graphs can also be displayed in R (See *.R files in the folder)
********************
```

2. Run postGSf90 including an option to calculate variance based on a window of 20 SNPs and an option to generate Manhattan plots. Use the default linear weight and the nonlinearA weight. Check the output files and compare results.

ssgblup.par

```
# BLUPF90 parameter file created by RENUMF90
DATAFILE
 ../renf90.dat
NUMBER_OF_TRAITS
           1
NUMBER_OF_EFFECTS
           2
OBSERVATION(S)
    1
WEIGHT(S)

EFFECTS: POSITIONS_IN_DATAFILE NUMBER_OF_LEVELS
 2          2 cross
 3      12010 cross
RANDOM_RESIDUAL VALUES
  0.60000
 RANDOM_GROUP
      2
 RANDOM_TYPE
 add_an_upginb
 FILE
../renadd02.ped
(CO)VARIANCES
  0.40000
OPTION SNP_file ../snp3.2k
OPTION map_file ../mrkmap.txt
OPTION no_quality_control
OPTION saveGInverse
OPTION saveA22Inverse
```

**NONLINEAR WEIGHTS**

```
# NONLINEAR WEIGHTS

mkdir non_linear ; cd non_linear
# Run blupf90 to save Gi, A22i and obtain solutions

cp ../renf90.par/renf90.par ssgblup.par .
echo ssgblup.par | blupf90 | tee blup.log
```

2. Run postGSf90 including an option to calculate variance based on a window of 20 SNPs and an option to generate Manhattan plots. Use the default linear weight and the nonlinearA weight. Check the output files and compare results.

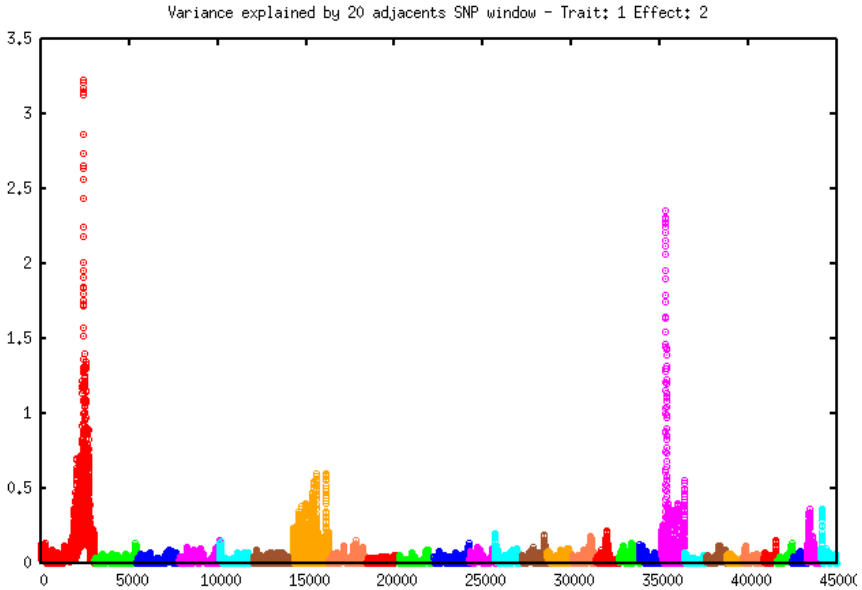
**NONLINEAR WEIGHTS**

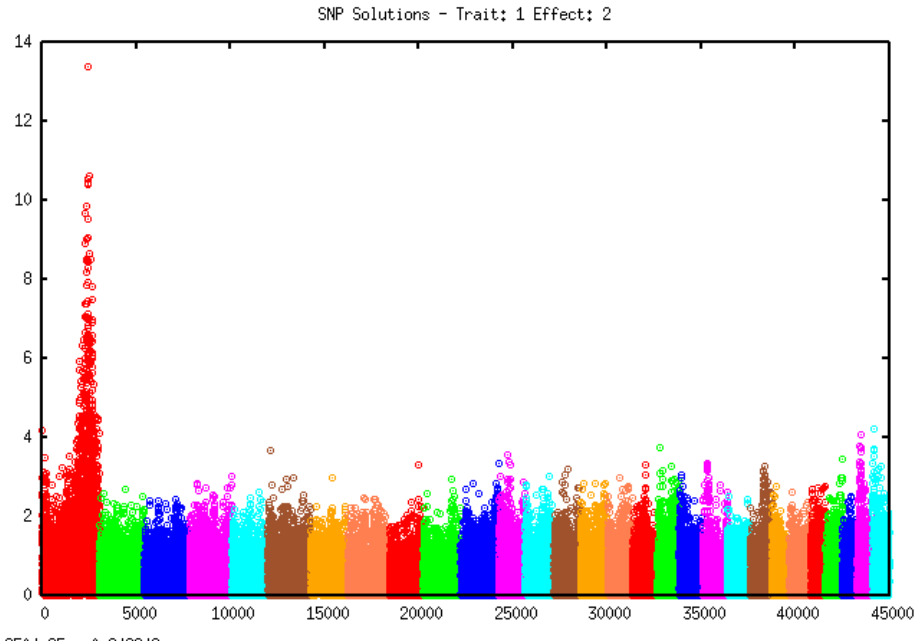
postgs_lin.par

```
# BLUPF90 parameter file created by RENUMF90
DATAFILE
 ../renf90.dat
NUMBER_OF_TRAITS
           1
NUMBER_OF_EFFECTS
           2
OBSERVATION(S)
    1
WEIGHT(S)

EFFECTS: POSITIONS_IN_DATAFILE NUMBER_OF_LEVELS
 2          2 cross
 3      12010 cross
RANDOM_RESIDUAL VALUES
  0.60000
 RANDOM_GROUP
      2
 RANDOM_TYPE
 add_an_upginb
 FILE
../renadd02.ped
(CO)VARIANCES
  0.40000
OPTION SNP file ../snp3.2k
OPTION map_file ../mrkmap.txt
OPTION no_quality_control
OPTION readGInverse
OPTION readA22Inverse
OPTION Manhattan_plot
OPTION windows_variance 20
OPTION which_weight nonlinearA
```

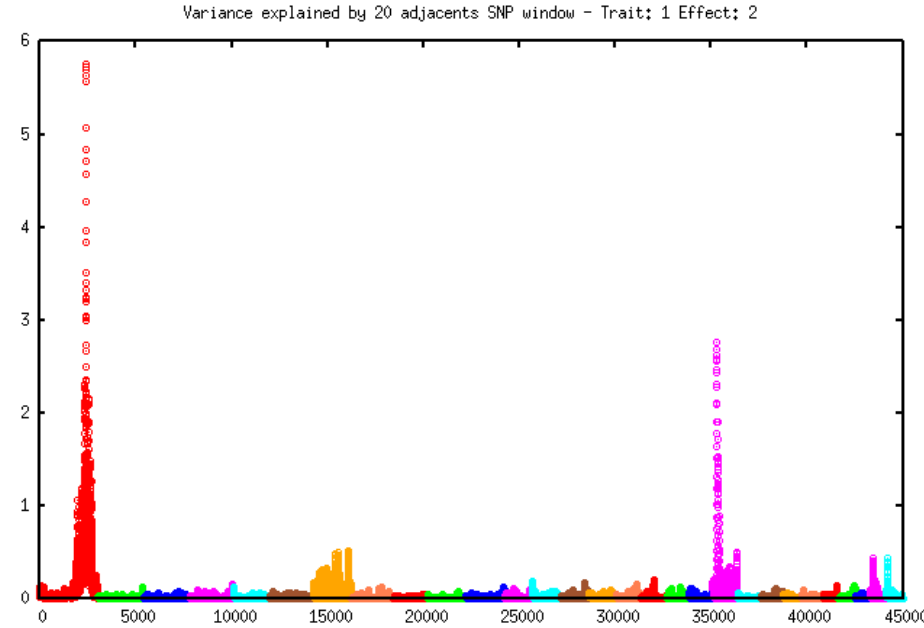
```
# Run postGSf90
cp ../ssgblup/renf90.par postgs_non.par
echo "OPTION Manhattan_plot" >> postgs_non.par
echo "OPTION windows_variance 20" >> postgs_non.par
echo "OPTION which_weight nonlinearA" >> postgs_non.par
echo postgs_non.par | postGSf90 | tee postgs.log


***** TIP *********
#The two graphs will be displayed automatically after postgsf90
#However, they can be displayed at any time using the command:
gnuplot Sftle2.gnuplot
gnuplot Vftle2.gnuplot
#The same graphs can also be displayed in R (See *.R files in the folder)
********************
```

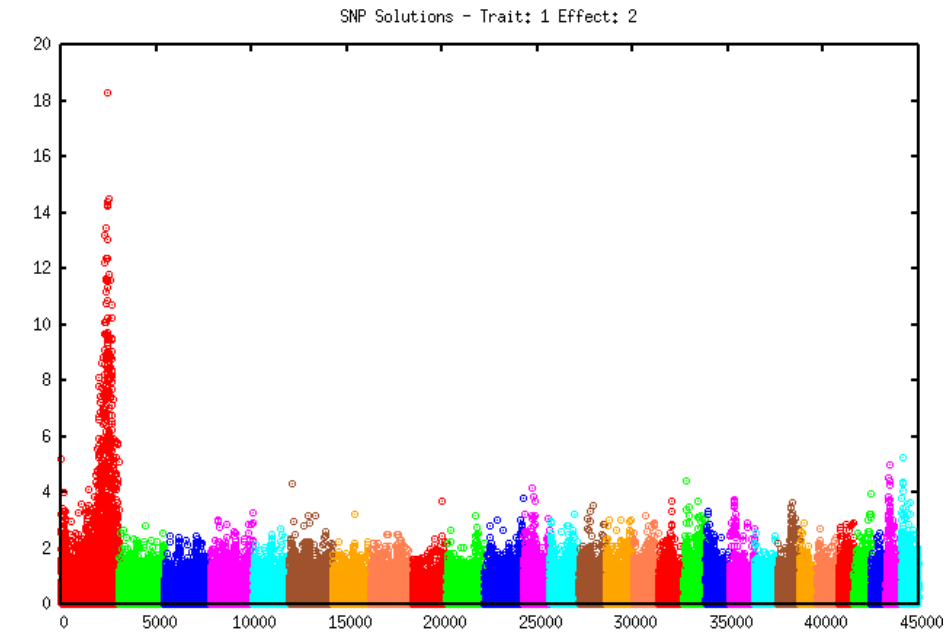
# LINEAR WEIGHTS




# NONLINEAR WEIGHTS



# Block 2

3. Iterative WssGBLUP: The weights computed in exercise 2 can be used to construct a weighted genomic relationship matrix $\mathbf{G_w} = \frac{\mathbf{ZDZ'}}{2\sum p_i(1-p_i)}$, and this matrix is used to compute new GEBV assuming SNP explain different proportions of variance. In this way, ssGBLUP becomes a weighted ssGBLUP (WssGBLUP). The WssGBLUP is an iterative method, where weights are used in several rounds, until there is no more change between weights in iteration t and t-1. Usually, 3 to 5 rounds are enough to obtain convergence. Run two rounds of `blupf90` and `postGSf90` for both linear and nonlinearA methods. Compare Manhattan plots and maximum variance explained.

   Hint 1: `blupf90` and `postGSf90` read a file with weights for each SNP if `OPTION weightedG file_name` is used. By default, all SNP have the same weight, so this file is actually a vector of dimension N x 1; where N is the number of SNP. If there are 50,000 SNP, the following command will create a vector of dimension 50,000 x 1:

```
awk 'BEGIN {for (i==1;i<50000;i++) print 1}' > w.txt
```

   Hint 2: updated weights are in column 7 of snp_sol (output from `postGSf90`). Use the following Unix command to create the new weight file (dimension N x 1) to be used in the second round of `blupf90` and `postGSf90`.

```
awk '{if ($1==1) print $7}' snp_sol > w
```

3. Iterative WssGBLUP: The weights computed in exercise 2 can be used to construct a weighted genomic relationship matrix $\mathbf{G_w} = \frac{\mathbf{ZDZ'}}{2\sum p_i(1-p_i)}$, and this matrix is used to compute new GEBV assuming SNP explain different proportions of variance. In this way, ssGBLUP becomes a weighted ssGBLUP (WssGBLUP). The WssGBLUP is an iterative method, where weights are used in several rounds, until there is no more change between weights in iteration t and t-1. Usually, 3 to 5 rounds are enough to obtain convergence. Run two rounds of `blupf90` and `postGSf90` for both linear and nonlinearA methods. Compare Manhattan plots and maximum variance explained.

```
# Run one more iteration, updating the weights
cd ../linear
#Making copies of files
cp solutions solutions_1
cp snp_sol snp_sol_1
cp chrsnp chrsnp_1

#Updating the weights:
awk 'NR>1 {print $7}' snp_sol > W
echo "OPTION weightedG W" >> ssgblup.par
echo "OPTION weightedG W" >> postgs_lin.par

echo ssgblup.par | blupf90 | tee ssgblup.log
echo postgs_lin.par | postGSf90 | tee postgs.log


cd ../non_linear
#Making copies of files
cp solutions solutions_1
cp snp_sol snp_sol_1
cp chrsnp chrsnp_1
#Updating the weights:
awk 'NR>1 {print $7}' snp_sol > W

echo "OPTION weightedG W" >> ssgblup.par
echo "OPTION which_weight nonlinearA" >> postgs_non.par

echo "OPTION weightedG W" >> postgs_non.par
echo "OPTION which_weight nonlinearA" >> postgs_non.par

echo ssgblup.par | blupf90 | tee ssgblup.log
echo postgs_non.par | postGSf90 | tee postgs.log
```

W is a column of weights

W is a column of weights

3. Iterative WssGBLUP: The weights computed in exercise 2 can be used to construct a weighted genomic relationship matrix $\mathbf{G_w} = \frac{\mathbf{ZDZ'}}{2\sum p_i(1-p_i)}$, and this matrix is used to compute new GEBV assuming SNP explain different proportions of variance. In this way, ssGBLUP becomes a weighted ssGBLUP (WssGBLUP). The WssGBLUP is an iterative method, where weights are used in several rounds, until there is no more change between weights in iteration t and t-1. Usually, 3 to 5 rounds are enough to obtain convergence. Run two rounds of `blupf90` and `postGSf90` for both linear and nonlinearA methods. Compare Manhattan plots and maximum variance explained.

```
#Example of iteration script for WssGBLUP

#Use a bash file for running this loop in the server
awk 'BEGIN { for (i==1;i<45000;i++) print 1}' > W
for i in {1..2}
do
echo ssgblup.par | blupf90 | tee ssgblup.log_$i
cp solutions solutions_$i
echo postgs.par | postGSf90 | tee postgs.log_$i
cp snp_sol snp_sol_$i
cp chrsnp chrsnp_$i
cp W W_$i
cp Sftle2.R plots/Sftle2_$i.R
cp Vftle2.R plots/Vftle2_$i.R
awk 'NR>1 {print $7}' snp_sol > W
done
```

Use a loop!