EXERCISE. SNP SOLVERS IN GENOMIC SELECTION

Andrés Legarra,
UR631 SAGA, INRA, 31326 Castanet Tolosan, France
andres.legarra at toulouse.inra.fr

The objective of this exercise is to fit SNP models using « snp-based » models (BLUP_SNP, BayesCPi, Bayessian Lasso) and software GS3.

This software can be found in http://snp.toulouse.inra.fr/~alegarra where the manual can be found.

Copy directory `/home/ads-guest3/andres/ex_qmsim`
Go to `ex_qmsim/2.` Take a look at files.
I have run QmSim and done a bit of clean up (don't do it yourself today!):

```
QMSim GenRes_example.prm
$ ./cleaningAfterQmSim.sh
```

This involves putting everything in UGA format, selecting 1,800 individuals from 28,800 and cleaning up homozygous loci. We also split individuals in training and validation; training have phenotypes while validation have not (they have 0's as phenotypes). We have true breeding values in the data files generated (in column 14).

Determine pedigree-based variance components

Use `remlf90` with the parameter file `p1_pedigreeonly.blupf90.par`. It gives σ^2_g=0.27 of genetic variance (simulated was 0.3), and σ^2_e=0.72. So we will use this for further analysis.

BLUP_SNP

The BLUP_SNP (Random Regression BLUP, GBLUP, BLUP in Meuwissen et al., 2001) assumes variance components are known. Therefore, we need to determine the variance components, and in particular the SNP variance for BLUP_SNP (this is simply called "BLUP" in GS3 manual). According to Gianola et al. 2009, this is (under some assumptions): $\sigma^2_a \approx \frac{\sigma^2_g}{2\sum p_i q_i}$, then we need to find out $2\sum p_i q_i$. This can be obtained using the "freq" file (a file with frequencies for each SNP) that was generated by me. For instance, you can compute it using R, or using awk:

```
$awk 'BEGIN{sum2pq=0}{sum2pq+=2*$1*(1-$1)}END{print sum2pq}' freq
```

The number $\sum p_i q_i$ will appear in the standard output of GS3 as well, look for it!!

This gives $2\sum p_i q_i = 8606.2$ so $\sigma^2_a \approx \frac{\sigma^2_g}{2\sum p_i q_i} = \frac{0.27}{8606.2} = 3.13727E-05$. This is a small number, but remember it goes multiplied by >24000 loci.

So let's launch GS3:

```
../gs3/gs3 p1_blup_snp.par
```
It generates a file with `solutions` (including SNPs) and another with EBV's in `p1_blup_snp.par_EBVs`.

Let's see in R the accuracy comparing the EBV's with the TBV, which are in column 14.
Open an R session:

```
 a=read.table("p1_blup_snp.par_EBVs",header=T)
 summary(a)
 b=read.table("p1_data_T.txt",header=F)
 summary(b)
 cor(a$EBV_overall,b$V14)
[1] 0.8156112
```

Which is actually pretty good, but these guys were in the training file. Now:

PREDICT the validation file using `p1_blup_snp.predict`.
This has to take the same format in the data file as the preceding analysis so that GS3 can compute things in the correct order. A complication is that all levels of all effects that exist in `predict` have to exist in `par` as well. E.g., if you do cross-validation and the whole data set has 1500 levels of herd then you need to "declare" the 1500 levels in both files. You might compare `p1_blup_snp.predict` and `p1_blup_snp.par`.

So we execute

```
../gs3/gs3 p1_blup_snp.predict
```

This creates a file called `predictions` and the EBV file as well.
Now let's check the quality of the prediction:

```
 a=read.table("p1_blup_snp.predict_EBVs",header=T)
 b=read.table("p1_data_V.txt",header=F)
 cor(a$EBV_overall,b$V14)
[1] 0.5271538
```

which is good but not as good as for the training data, as could be expected.
Let's take a look at SNP results:

```
 a=read.table("solutions",header=TRUE)
 summary(a)
 snp=a[a$effect==2,]
 summary(snp)
 plot(snp$solution)
```

No "large" snp can be easily spotted.

- Other diagnostics should be done. What is the bias and inflation of EBVs? This can be checked by fitting a linear model

$$TBV = a + b\,EBV$$

, i.e., in R:

```
 summary(lm(b$V14 ~ a$EBV_overall))
```

Also, method closest to "best" should have minimum MSE, that can be easily computed as

```
 mean((b$V14 - a$EBV_overall)**2)
```

BAYESIAN ANALYSES BY GIBBS SAMPLING

Later analyses are in directories `BayesianLasso, BayesC, BayesCPi`

NOTE: these analysis might be slow because they are MCMC. You might reduce the number of iterations in the parameter file from 10000 to 4000 for instance, results will be usually poorer.

BayesC

This estimates the variances as well, and from a BLUP_SNP file it is straightforward: change "method" BLUP to VCE (but it will take much longer, and you need to verify convergence of the MCMC, etc). The parameter file is `p1_BayesC.par`. It usually gives similar results in livestock populations (but not in the mice data set) as the approximate identity for the variances is rather good. In this actual analysis, the estimate of σ_a^2 does change. So finally, The correlation(TBV,EBV) is 0.80 for the training set and 0.54 for the validation one (just slightly better). To compute the correlation, run the `predict` file and do as above, changing appropriately the file names:

```
a=read.table("p1_BayesC.predict_EBVs",header=T)
b=read.table("../p1_data_V.txt",header=F)
cor(a$EBV_overall,b$V14)
```

BayesCPi

We will fix π to 999/1000, i.e., only 1 SNP every 1000 is supposed to "have" an effect. (There are 750 QTLs simulated and 24,000 SNPs so this is completely wrong). Parameter π can be equally estimated but in our experience π is very much confounded with σ_a^2, so when one increases the other decreases. A good guess, or starting value for σ_a^2 is

$\sigma_a^2 \approx \frac{\sigma_g^2}{(1-\pi)2\sum p_i q_i} = 3.13727E-02$. (Note that this σ_a^2 is defined for those SNPs *which are not 0*).

The parameter file is modified by stating "mixture TRUE" in the last line. We also fix the proportions in the *a priori* beta distribution so that π will be fixed in practice to 999/1000, as follows:

```
A PRIORI a
1d8 999d8
...
USE MIXTURE
T
```

Note that GS3 has the opposite notation for π in the documentation.
We launch it as usual

```
../gs3/gs3 p1_BayesCPi.par
```

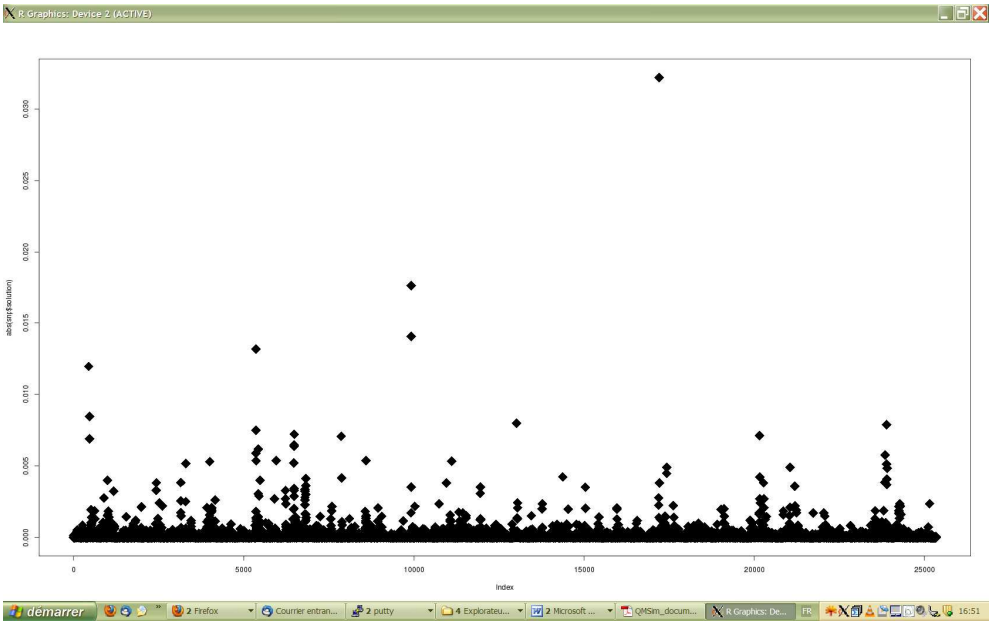
In the output you can see the number of SNPs “drawn” at an iteration: “includeda”. In the parameter file I have put 10,000 iterations *but this is too little.* We usually put 100,000 iterations or more. Imagine: it should ideally explore all combinations of 24 SNP in a space of some 24,000 SNPs. (Hence, in my opinion this is a source of inefficiency and other options such as BLUP_SNP or non-linear Lasso, ElasticNet or VanRaden’s nonlinearA should be better, specially for very large SNP data.)

With these caveats, accuracy in the validation data set is 0.70, and 0.48 in the training data set. Not a very good accuracy but possibly fixing π=0.999 was not a good idea. A value 0.99 or estimating π is possibly a better idea.

However, if we look at individual SNP effects using R there are interesting things:

```
a=read.table("solutions",header=TRUE)
summary(a)
snp=a[a$effect==2,]
summary(snp)
plot(snp$solution)
plot(abs(snp$solution),cex=3,pch=18)
```

Some very large SNP can spotted at position ~17204, this is possibly a large QTL location. We have found that this way of fitting BayesCPi is as good as other methods to spot QTL locations.



Bayesian Lasso (BL)

This is rather similar to BayesC as well, but we *don’t* use the “mixture” option and we do use an `OPTION BayesianLasso Tibshirani` , which corresponds to BL2Var in Legarra et al. 2011 (option `ParkCasella` has been planned yet not executed ! volunteers welcome). The starting value for lambda is deduced from the starting value for σ_a^2 (for instance, $\sigma_a^2 \approx \frac{\sigma_g^2}{2\sum p_i q_i}$) as $\lambda^2 = \frac{2}{\sigma_a^2}$ because of the equivalence $\sigma_g^2 \approx 2\sum p_i q_i \frac{2}{\lambda^2}$ (Legarra

et al., 2011). The BL is generally faster than BayesCPi because it does not compute any likelihood. Also, does not require this "exploration" so in my opinion mixing should be much better.
The accuracy with training is 0.79, whereas with validation is 0.54.

Things to do

- Other diagnostics should be done. What is the bias and inflation of EBVs? This can be checked by fitting a linear model

$$TBV = a + b\,EBV$$

, i.e., in R:

```
summary(lm(b$V14 ~ a$EBV_overall))
```

Also, method closest to "best" should have minimum MSE, that can be easily computed as

```
mean((b$V14 - a$EBV_overall)**2)
```

- You can absolutely take a look at the file `var` with samples of variance components, to check convergence. The simplest way to do is, using R:
  1) discard visually burn-in iterations
  2) with the remaining samples, compare the results using the first half vs. using the second half. If the chain has converged to the posterior distribution, results should be similar.
- You can change whatever parameters (π, variances) and compare the results.
- You should take a look at the distribution of estimates of SNP effects in the file `solutions`.
- If you read the documentation, you'll find out that GS3 can estimate the genetic parameters with pedigree (no SNPs whatsoever) and also separate genetic variance due to SNPs from genetic variance due to pedigree.