Practical 5: Quantitative genetics and heritability estimation

Jerome Goudet and Bruce Weir

2021-07-16

```
library(gaston)
library(hierfstat)
library(JGTeach)
```

Heritability of (simulated) traits from (simulated) panmictic populations

1. We will use again the data we generated during the first practical, where we used `ms` to simulate data from a panmictic population. You may remember we simulated for 500 individuals 20 chromosomes 1 Megabase long.

— use function `JGTeach::make.traits` to simulate a trait with a heritability of 0.8, with 100 causal loci and using all possible SNPs as possible causal loci. Explore the structure of the object generated (you might also want to look at the function help page). Plot a histogram of the distribution of the trait you have generated. Plot the phenotypic value of the trait against its breeding value. Explain the difference between h^2 and $\hat{h^2}$

```
pan<-ms2bed("pan.txt")

#command issued to generate the data
readLines("pan.txt",n=1L)
```

```
## [1] "ms 1000 20 -t 200 -r 200 1000000 -p 8 "
```

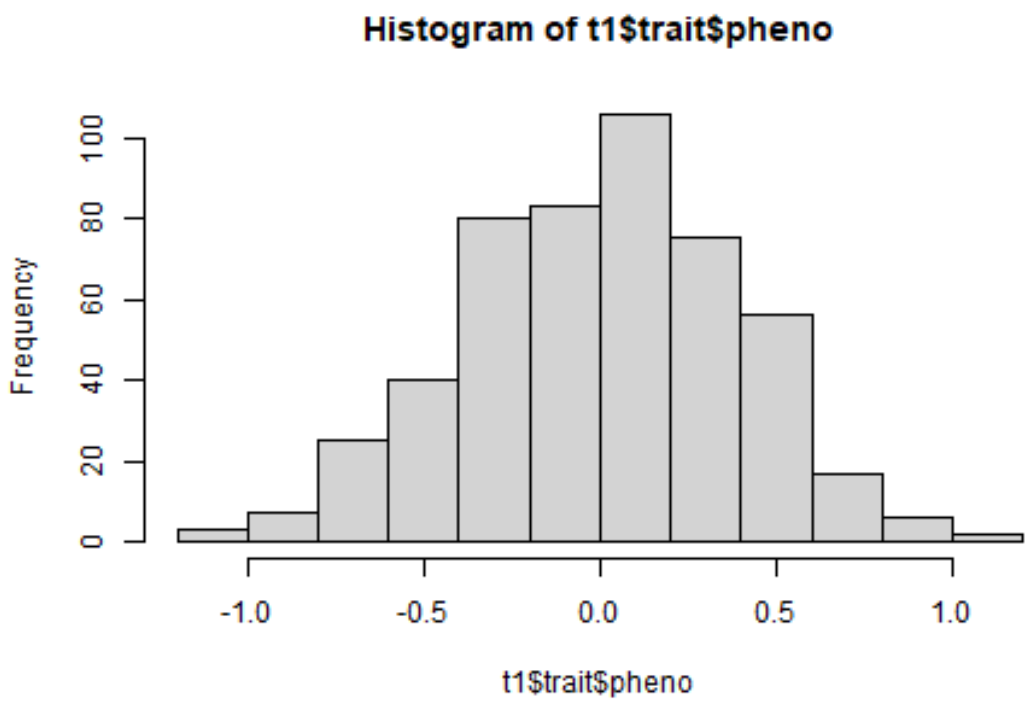
```
set.seed(87)

t1<-make.traits(pan,h2=0.8,n.causal=100,minp=0)

str(t1)
```

```
## List of 4
##  $ h2    : num 0.8
##  $ h2hat : num 0.81
##  $ trait :'data.frame':  500 obs. of  2 variables:
##   ..$ BV   : num [1:500] -0.0149 -0.2541 -0.2405 0.0505 -0.363 ...
##   ..$ pheno: num [1:500] 0.0936 -0.4572 -0.1738 0.078 -0.6318 ...
##  $ causal:'data.frame':  100 obs. of  3 variables:
##   ..$ chr: int [1:100] 1 13 2 16 2 4 8 14 18 16 ...
##   ..$ id : chr [1:100] "M_1054" "M_19098" "M_2783" "M_23440" ...
##   ..$ efs: num [1:100] -0.000724 -0.142978 -0.008772 -0.110842 -0.177222 ...
```

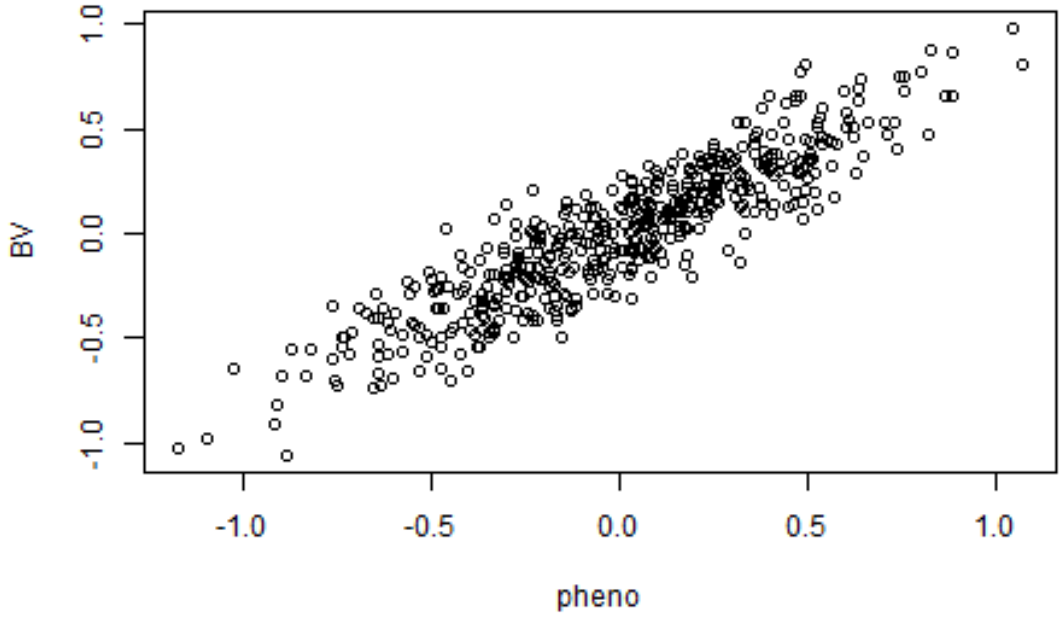
```
t1$h2hat
```

```
## [1] 0.8095129
```

```
hist(t1$trait$pheno)
```




```
with(t1$trait,plot(pheno,BV))
```



```
with(t1$trait,cor(BV,pheno)^2)
```

```
## [1] 0.8155893
```

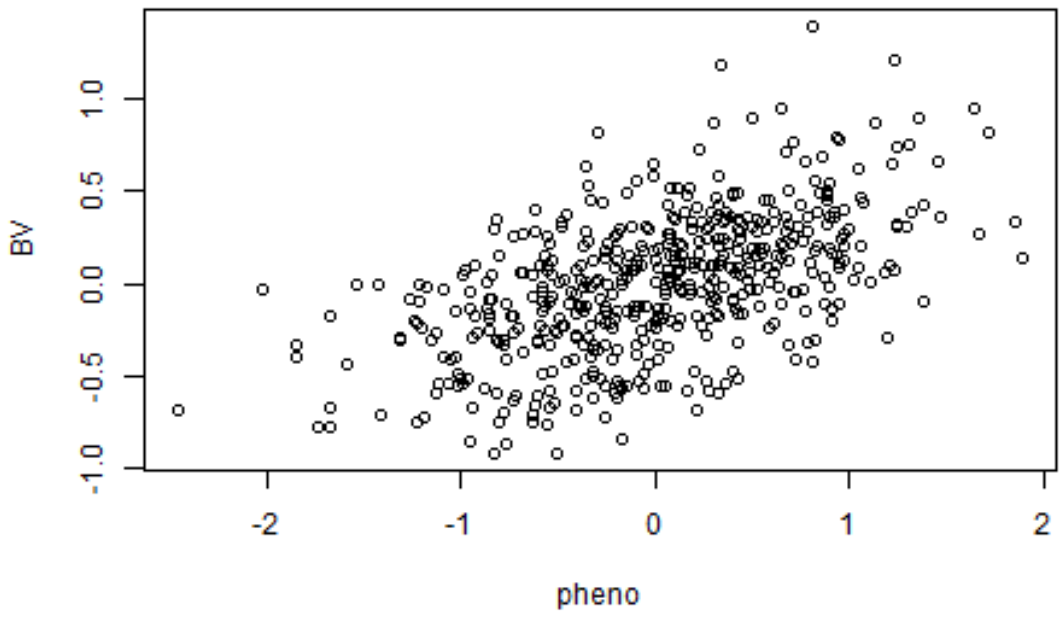
```
h2 is the theoretical heritability we aimed for,
h2hat is the realized one, from the sampling process of individuals.

The distribution of the trait is bell shaped, centered on 0.
The list produced also contain element trait, with breeding values
(the genetic contribution to the trait), and phenotype.
And element causal, describing the chromosome, name and effect size of the causal loci.
The relation between phenotype and breeding value is tight, because h2=0.8.
```

— Create a second trait with the same characteristics but a heritability of 0.3. Plot again breeding value against phenotype and explain the difference from the previous trait (it might be helpful to quantify the spread)

```
t2<-make.traits(pan,h2=0.3,n.causal=100,minp=0)
t2$h2hat
```

```
## [1] 0.2920886
```

```
with(t2$trait,plot(pheno,BV))
```




```
with(t2$trait,cor(BV,pheno))^2
```

```
## [1] 0.2992219
```

```
For the second trait, the heritability is lower,
and the spread of phenotype around breeding value is larger
```

Next we will use a linear mixed model to estimate the heritability of this trait. This requires a trait and a GRM. We will thus estimate heritability with the 3 GRMS we discussed this morning.

2. Compute heritabilities for trait 1 and 2 using the 3 GRMs and function `gaston::lmm.aireml`. Are the heritability estimates from the 3 GRMS similar ?

```
#first compute the GRMs

#allele sharing GRM
GRM.as.pan<-hierfstat::kinship2grm(hierfstat::beta.dosage(pan,inb=TRUE))

#Kc0 GRM
GRM.c0.pan<-2*(JGTeach::Kc0(pan))

#Standard GRM
std.pan<-gaston::GRM(pan)

#extract heritability from an lmm.aireml object

herit<-(function(x) x$tau/(x$tau+x$sigma2))

#Trait 1

#value to be estimated
t1$h2hat
```

```
## [1] 0.8095129
```

```
#estimations
herit(rAS<-gaston::lmm.aireml(t1$trait$pheno,K=GRM.as.pan,verbose=FALSE))
```

```
## [1] 0.9229796
```

```
herit(rC0<-gaston::lmm.aireml(t1$trait$pheno,K=GRM.c0.pan,verbose=FALSE))
```

```
## [1] 0.9229086
```

```
herit(rStd<-gaston::lmm.aireml(t1$trait$pheno,K=std.pan,verbose=FALSE))
```

```
## [1] 0.9999926
```

```
# Trait 2

#value to be estimated
t2$h2hat
```

```
## [1] 0.2920886
```

```
#estimations
herit(lmm.aireml(t2$trait$pheno,K=GRM.as.pan,verbose=FALSE))
```

```
## [1] 0.3827383
```

```
herit(lmm.aireml(t2$trait$pheno,K=GRM.c0.pan,verbose=FALSE))
```

```
## [1] 0.3825026
```

```
herit(lmm.aireml(t2$trait$pheno,K=std.pan,verbose=FALSE))
```

```
## [1] 0.6107762
```

```
GRM computed from Kas or Kco give very similar answers,
and close to realized h2, as expected.

Standard GRM gives a large overestimation for the two traits
```

— A commonly seen recommendation for the standard GRM is to filter on low MAF. Re-estimate heritabilities using the standard GRM filtered on $maf > 0.05$

```
std.pan.maf05<-GRM(pan[,pan@snps$maf>=0.05])

herit(lmm.aireml(t1$trait$pheno,K=std.pan.maf05,verbose=FALSE))
```

```
## [1] 0.8932641
```

```
herit(lmm.aireml(t2$trait$pheno,K=std.pan.maf05,verbose=FALSE))
```

```
## [1] 0.3928383
```

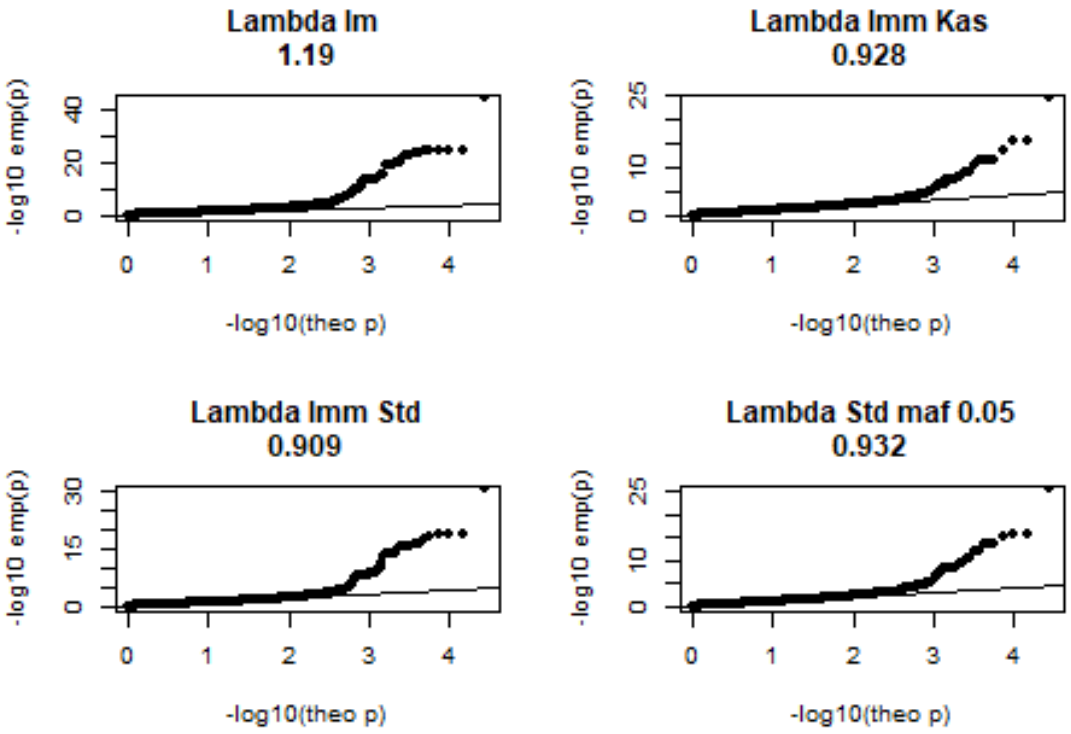
```
When we estimate standard GRM with filtering on MAF,
things are getting better
```

— [optional] using `gaston::association.test`, perform GWAS using a simple linear model and using a linear mixed model with the fours GRMs estimated before and estimate the genomic inflation factor λ

```
p.lm<-association.test(pan,Y=t1$trait$pheno,method="lm")
```

```
## Warning in association.test(pan, Y = t1$trait$pheno, method = "lm"): Method =
## "lm" force test = "wald"
```

```
pAS.lmm<-association.test(pan,Y=t1$trait$pheno,method="lmm",K=GRM.as.pan,verbose=FALSE)
pc0.lmm<-association.test(pan,Y=t1$trait$pheno,method="lmm",K=GRM.c0.pan,verbose=FALSE)
pstd.lmm<-association.test(pan,Y=t1$trait$pheno,method="lmm",K=std.pan,verbose=FALSE)
pstdmaf05.lmm<-association.test(pan,Y=t1$trait$pheno,method="lmm",K=std.pan.maf05,verbose=FALSE)

# get genomic inflation factor
get.lambda<-function(x) {xchisq<-qchisq(1-x,1);median(xchisq)/qchisq(0.5,1)}

#plot quantile quantile plot of p-values
pval.qplot<-function(x,...){
  n<-length(x)
  plot(-log10(1:n/n),-log10(sort(x)),
       xlab="-log10(theo p)",ylab="-log10 emp(p)",pch=16,...);abline(c(0,1))
}

par(mfrow=c(2,2))
pval.qplot(p.lm$p,main=paste0("Lambda lm \n",round(get.lambda(p.lm$p),digits=3)))
pval.qplot(pAS.lmm$p,main=paste0("Lambda lmm Kas\n",round(get.lambda(pAS.lmm$p),digits=3)))
pval.qplot(pstd.lmm$p,main=paste0("Lambda lmm Std\n",round(get.lambda(pstd.lmm$p),digits=3)))
pval.qplot(pstdmaf05.lmm$p,main=paste0("Lambda Std maf 0.05\n",round(get.lambda(pstdmaf05.lmm$p),dig
```



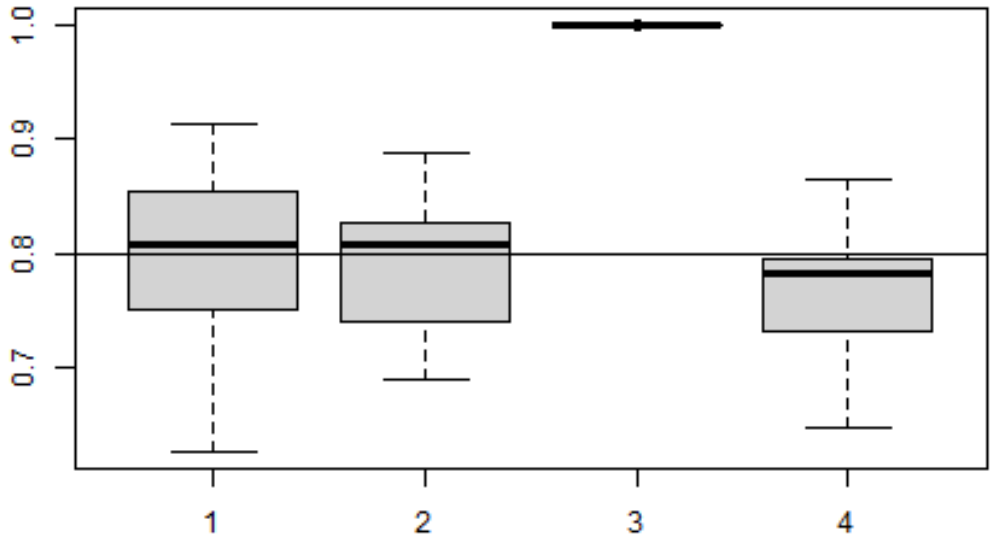

```
par(mfrow=c(1,1))
```

Even in panmictic population, the linear model tends to give to many significant p-values. the genomic inflation factor is well controlled with a GRM based on allele sharing and on C0, whereas the standard GRM over corrects. Filtering on MAF for teh standard GRM improves things.

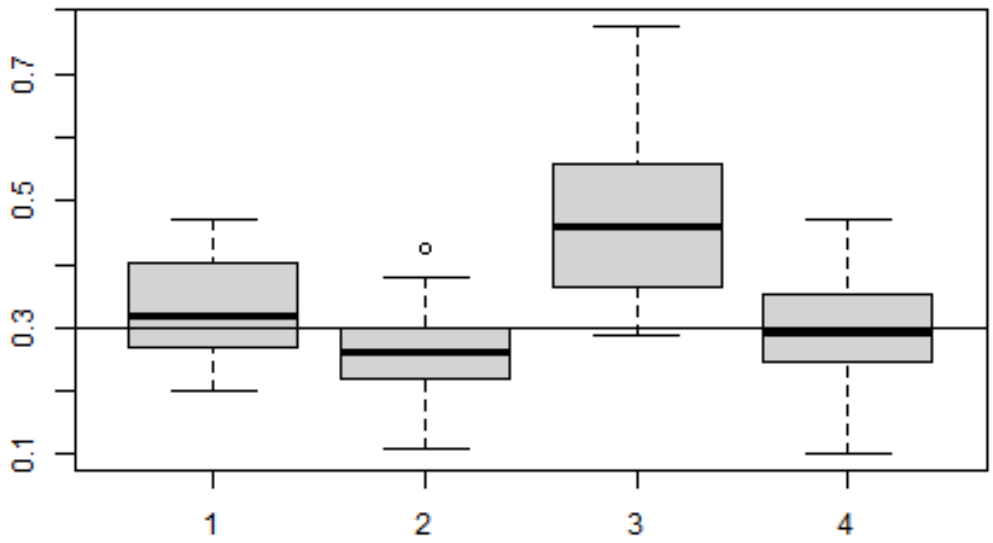
— [optional] Evaluate heritabilities for other traits. Is there a pattern emerging ?

```
# The get.herit function creates a trait and estimate
# its heritability with lmm and the given GRM

get.herit<-function(bed,h2=0.8,n.causal=100,minp=0,GRM=GRM,...){

    herit<-function(x) x$tau/(x$tau+x$sigma2)

  tx<-make.traits(bed=bed,h2=h2,n.causal=n.causal,minp=minp,...)
  est<-herit(lmm.aireml(Y=tx$trait$pheno,K=GRM,verbose=FALSE,...))
  c(tx$h2hat,est)
}


#for h2=0.8

h2.as<-replicate(20,get.herit(pan,GRM=GRM.as.pan))
h2.c0<-replicate(20,get.herit(pan,GRM=GRM.c0.pan))
h2.std<-replicate(20,get.herit(pan,GRM=std.pan))
h2.std.maf05<-replicate(20,get.herit(pan,GRM=std.pan.maf05))
```

```
boxplot(cbind(h2.as[2,],h2.c0[2,],h2.std[2,],h2.std.maf05[2,]))
abline(h=0.8)
```




```
#for h2=0.3

h2.as<-replicate(20,get.herit(pan,h2=0.3,GRM=GRM.as.pan))
h2.c0<-replicate(20,get.herit(pan,h2=0.3,GRM=GRM.c0.pan))
h2.std<-replicate(20,get.herit(h2=0.3,pan,GRM=std.pan))
h2.std.maf05<-replicate(20,get.herit(h2=0.3,pan,GRM=std.pan.maf05))

boxplot(cbind(h2.as[2,],h2.c0[2,],h2.std[2,],h2.std.maf05[2,]))
#boxplot(cbind(t(h2.as),t(h2.c0),t(h2.std),t(h2.std.maf05)))
abline(h=0.3)
```




```
Replicating this over 20 traits shows that Kas and Kc0 give very close,
and accurate estimates of heritability.
Standard does not. filtering on MAF helps
```

Heritability from simulated traits using (part of) 1000 genome data

We can redo the same exercice this time using the subset of data we have been using since the begininng of this module. The difference from the simulated data is we have population structure, admixture etc in the data set. What will be the effect of these on heritability estimates ?

3. simulate one trait with heritability 0.5 and driven by 10 causal loci with $MAF >= 0.01$ using the AMR samples from the 1000 genomes data, and estimate its heritability using the 4 GRMS we used previously (Allele sharing As ; c0 ; standard ; and standard filter on $maf \geq 0.05$). Discuss the results and their shortcomings

```
ch22<-read.VCF("chr22_Mb0_20.recode.vcf.gz")
```

```
## ped stats and snps stats have been set.
## 'p' has been set.
## 'mu' and 'sigma' have been set.
```

```
ch22.M<-readRDS("matching.ch22.RDS")

samp.desc<-read.table("ftp://ftp-trace.ncbi.nih.gov/1000genomes/ftp/release/20130502/integrated_call

AMR<-which(samp.desc$super_pop=="AMR")
ch22AMR<-ch22[AMR,]
ch22AMR<-ch22AMR[,ch22AMR@snps$maf>0]
ch22AMR.M<-ch22.M[AMR,AMR]
```

```
Mb<-mean(mat2vec(ch22AMR.M))

ch22AMR.Kas<-(ch22AMR.M-Mb)/(1-Mb)
ch22AMR.Kc0<-JGTeach::Kc0(ch22AMR.M,matching=TRUE)
ch22AMR.std.GRM<-gaston::GRM(ch22AMR)
ch22AMR.std.GRM.maf05<-gaston::GRM(ch22[AMR,ch22@snps$maf>=0.05])

set.seed(13)

t1AMR<-make.traits(ch22AMR,n.causal=10,h2=0.5,minp=0.01)

t1AMR$h2hat
```

```
## [1] 0.4526661
```

```
#estimations
herit(rASAMR<-gaston::lmm.aireml(t1AMR$trait$pheno,K=2*ch22AMR.Kas,verbose=FALSE))
```

```
## [1] 0.3094306
```

```
herit(rC0AMR<-gaston::lmm.aireml(t1AMR$trait$pheno,K=2*ch22AMR.Kc0,verbose=FALSE))
```

```
## [1] 0.3091145
```

```
herit(rStdAMR<-gaston::lmm.aireml(t1AMR$trait$pheno,K=ch22AMR.std.GRM,verbose=FALSE))
```

```
## [1] 0.3340894
```

```
herit(rStdAMRmaf05<-gaston::lmm.aireml(t1AMR$trait$pheno,K=ch22AMR.std.GRM.maf05,verbose=FALSE))
```

```
## [1] 0.3145598
```

— Are estimates of heritability impacted by population structure ?

```
Not really, When using 1000 genome data, we observed the same thing:
Kas and Kc0 do well, standard does not, filtering may help
```

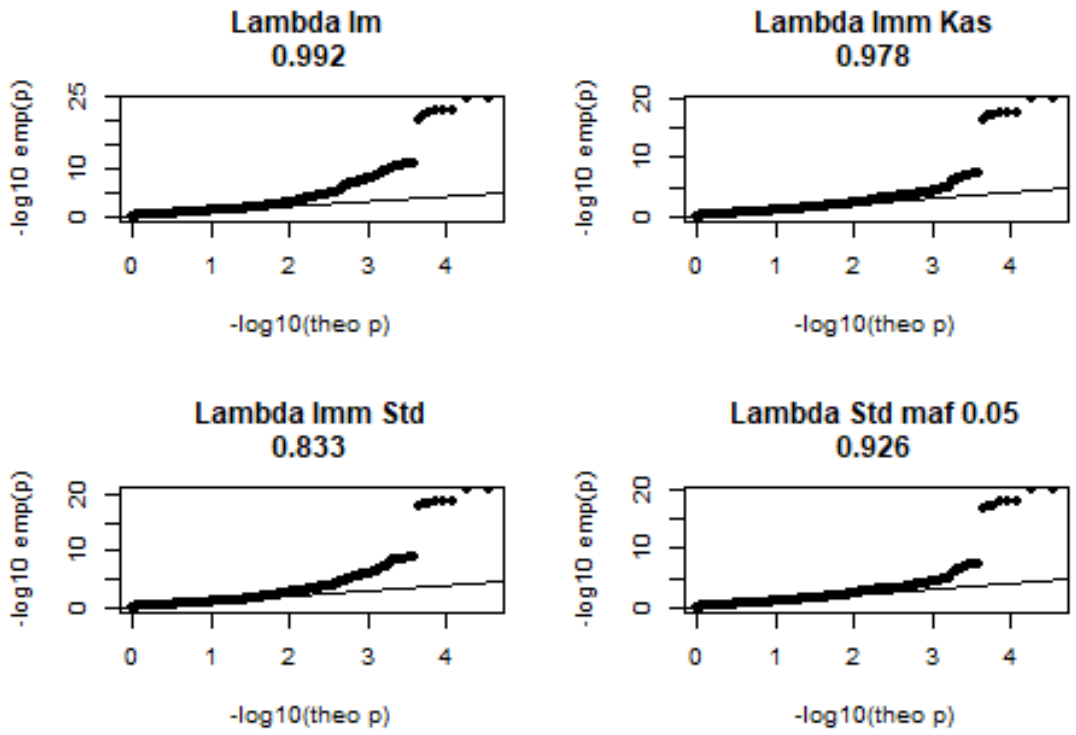
— [optional] Using the function `gaston::association.test`, perform GWAS on this trait, using either a linear model with no covariate or a linear mixed model with the 4 GRM we discussed ; plot the qqplot of the p-values, estimate the genomic correction factor for each and interpret the results.

```
nsnps<-dim(ch22AMR)[2]
p.lm<-association.test(ch22AMR,Y=t1AMR$trait$pheno,method="lm")
```

```
## Warning in association.test(ch22AMR, Y = t1AMR$trait$pheno, method = "lm"):
## Method = "lm" force test = "wald"
```

```
pAS.lmm<-association.test(ch22AMR,Y=t1AMR$trait$pheno,method="lmm",K=2*ch22AMR.Kas,verbose=FALSE)
pc0.lmm<-association.test(ch22AMR,Y=t1AMR$trait$pheno,method="lmm",K=2*ch22AMR.Kc0,verbose=FALSE)
pstd.lmm<-association.test(ch22AMR,Y=t1AMR$trait$pheno,method="lmm",K=ch22AMR.std.GRM,verbose=FALSE)
pstdmaf05.lmm<-association.test(ch22AMR,Y=t1AMR$trait$pheno,method="lmm",K=ch22AMR.std.GRM.maf05,ver

# get genomic inflation factor
get.lambda<-function(x) {xchisq<-qchisq(1-x,1);median(xchisq)/qchisq(0.5,1)}

#plot quantile quantile plot of p-values
```

```
pval.qplot<-function(x,...){
  n<-length(x)
  plot(-log10(1:n/n),-log10(sort(x)),
       xlab="-log10(theo p)",ylab="-log10 emp(p)",pch=16,...);abline(c(0,1))
}

par(mfrow=c(2,2))
pval.qplot(p.lm$p,main=paste0("Lambda lm \n",round(get.lambda(p.lm$p),digits=3)))
pval.qplot(pAS.lmm$p,main=paste0("Lambda lmm Kas\n",round(get.lambda(pAS.lmm$p),digits=3)))
pval.qplot(pstd.lmm$p,main=paste0("Lambda lmm Std\n",round(get.lambda(pstd.lmm$p),digits=3)))
pval.qplot(pstdmaf05.lmm$p,main=paste0("Lambda Std maf 0.05\n",round(get.lambda(pstdmaf05.lmm$p),dig
```



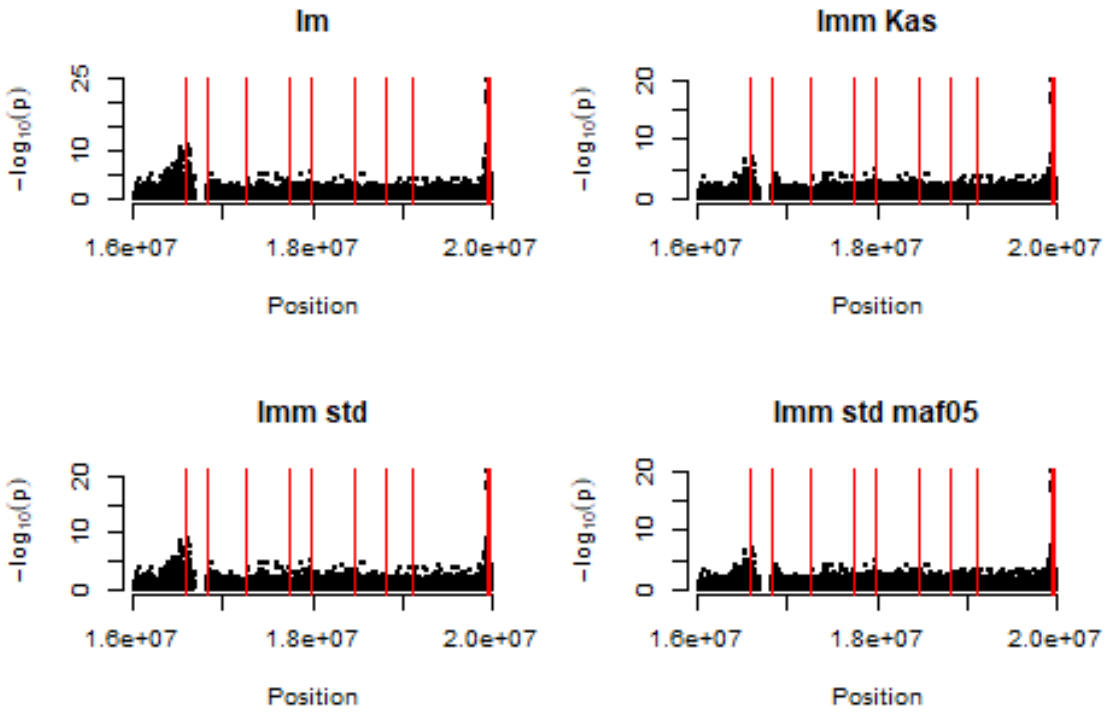

```
par(mfrow=c(1,1))
```

We observe here the same pattern as what we saw in the panmictic population, just amplified.

— [optional] Using the function `gaston::manhattan` produce manhattan plots of the p-values along the genome segment, and interpret the results

```
par(mfrow=c(2,2))
manhattan(x=data.frame(chr=ch22AMR@snps$chr,pos=ch22AMR@snps$pos,p=p.lm$p),
          thinning=FALSE,pch=16,cex=0.5,main="lm")
abline(v=ch22AMR@snps$pos[match(t1AMR$causal$id,pAS.lmm$id)],col="red")

manhattan(x=data.frame(chr=ch22AMR@snps$chr,pos=ch22AMR@snps$pos,p=pAS.lmm$p),
          thinning=FALSE,pch=16,cex=0.5,main="lmm Kas")
abline(v=ch22AMR@snps$pos[match(t1AMR$causal$id,pAS.lmm$id)],col="red")
manhattan(x=data.frame(chr=ch22AMR@snps$chr,pos=ch22AMR@snps$pos,p=pstd.lmm$p),
          thinning=FALSE,pch=16,cex=0.5,main="lmm std")
```

```
abline(v=ch22AMR@snps$pos[match(t1AMR$causal$id,pAS.lmm$id)],col="red")

manhattan(x=data.frame(chr=ch22AMR@snps$chr,pos=ch22AMR@snps$pos,p=pstdmaf05.lmm$p),
          thinning=FALSE,pch=16,cex=0.5,main="lmm std maf05")
abline(v=ch22AMR@snps$pos[match(t1AMR$causal$id,pAS.lmm$id)],col="red")
```




```
par(mfrow=c(1,1))
```

While the GWASs usually identify the regions of causal SNPs, the GWAS based on the linear model tend to find spurious regions, and thsi is also true to a lesser extent for the linear mixed model with the std GRM. Filtering on maf for this last GRM helps