Computer Practical Exercise using GCTA (with R)

Overview

Purpose

This exercise repeats the linear mixed model analysis from the previous exercise using the program GCTA instead of FaST-LMM. In addition, we use GCTA to estimate the heritability accounted for by all genotyped SNPs, and by various subsets of SNPs.

Methodology

We will use the linear mixed model approach implemented in GCTA.

Program documentation

GCTA documentation:

Documentation can obtained together with the GCTA program from:

http://cnsgenomics.com/software/gcta/

Data overview

As a reminder, we are using family data consisting of 498 individuals typed at 134,946 SNPs. All individuals have measurements of a quantitative trait of interest.

Appropriate data

Appropriate data for this exercise is genome-wide genotype data for individuals who are phenotyped for either a dichotomous trait or a quantitative trait of interest. GCTA is really designed for the analysis of apparently unrelated individuals, but in this case we will apply it to a set of related individuals, in order to compare the results with those we obtained previously for these individuals.

Instructions

Data files

We will use the same PLINK binary-file format files `quantfamdata.bed`, `quantfamdata.bim` and `quantfamdata.fam` used previously. We will also use R to create an additional phenotype file required by GCTA.

Step-by-step instructions

1. Create phenotype file in R

To start with, we will use R to create the phenotype file required by GCTA. Start R (by typing `R`) and create a new phenotype file from the .fam file by typing the following commands:

```
fam<-read.table("quantfamdata.fam", header=F)
pheno=data.frame(fam[,1:2],fam[,6])
write.table(pheno,file="phenos.txt",col.names=F,row.names=F,quote=F)
```

Take a look at the file `phenos.txt` that you just created, to check you understand it.

2. GCTA Analysis

To use GCTA to perform association analysis while allowing for relatedness between individuals, type:

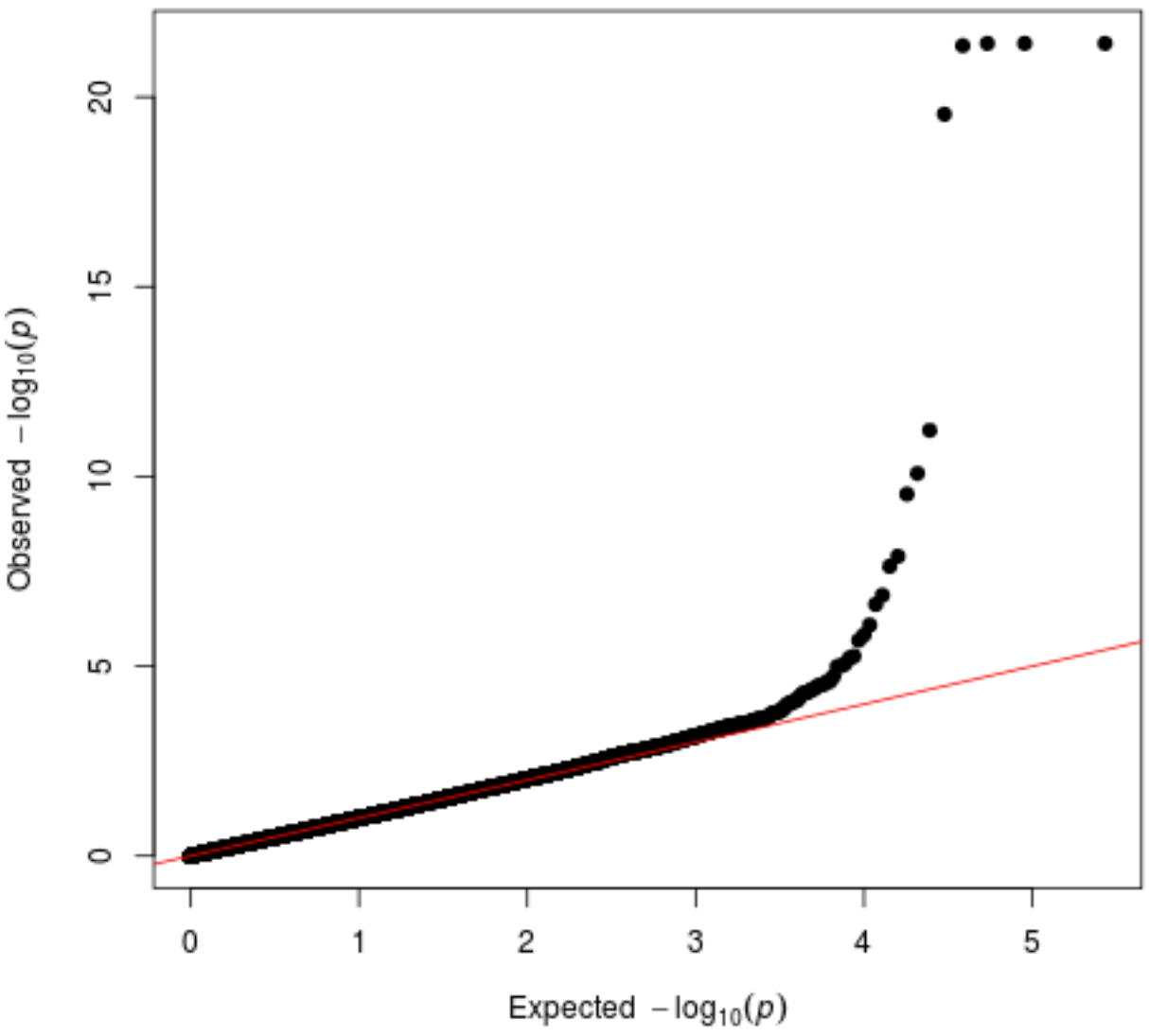
```
gcta64 --mlma --bfile quantfamdata --pheno phenos.txt --out GCTAresults
```

Here we use the `--mlma` option to tell GCTA to perform association analysis, we use the `--bfile` and `--pheno` options to tell GCTA which files to read in the genotype and phenotype data from, and we use `GCTAresults` as the stem name for the output files.

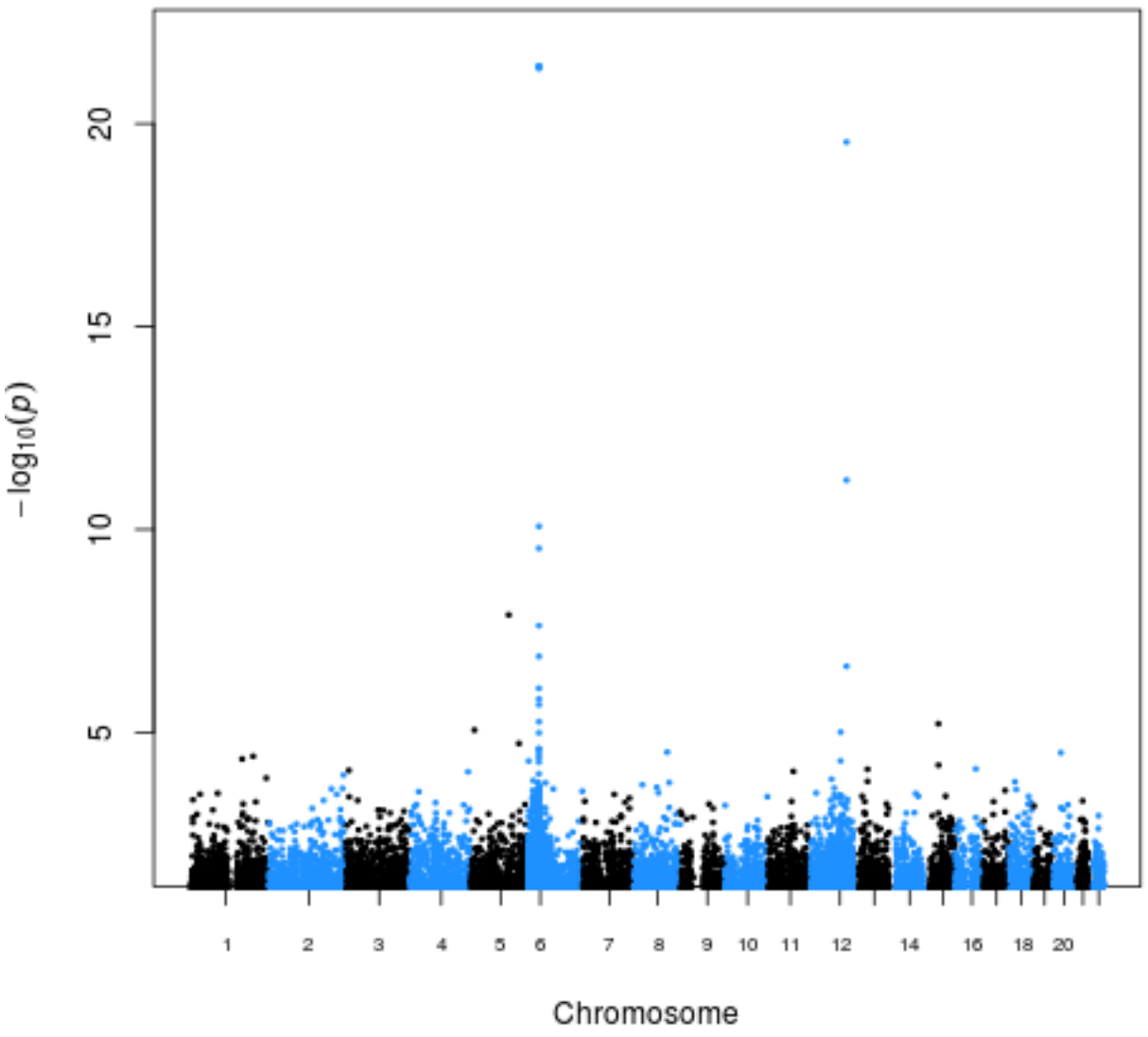
To calculate the genomic control inflation factor, and to produce QQ and Manhattan plots from the above analysis, you can use the following sequence of commands within R. (Make sure that you understand the commands - if not please ask an instructor).

```
source("qqmanHJCupdated.R")

res3<-read.table("GCTAresults.mlma", header=T)
head(res3)

chi<-(qchisq(1-res3$p,1))
lambda=median(chi)/0.456
lambda
```

```
new3<-data.frame(res3$SNP, res3$Chr, res3$bp, res3$p)
names(new3)<-c("SNP", "CHR", "BP", "P")
head(new3)

png("qq3.png")
qq(new3$P)
dev.off()
```



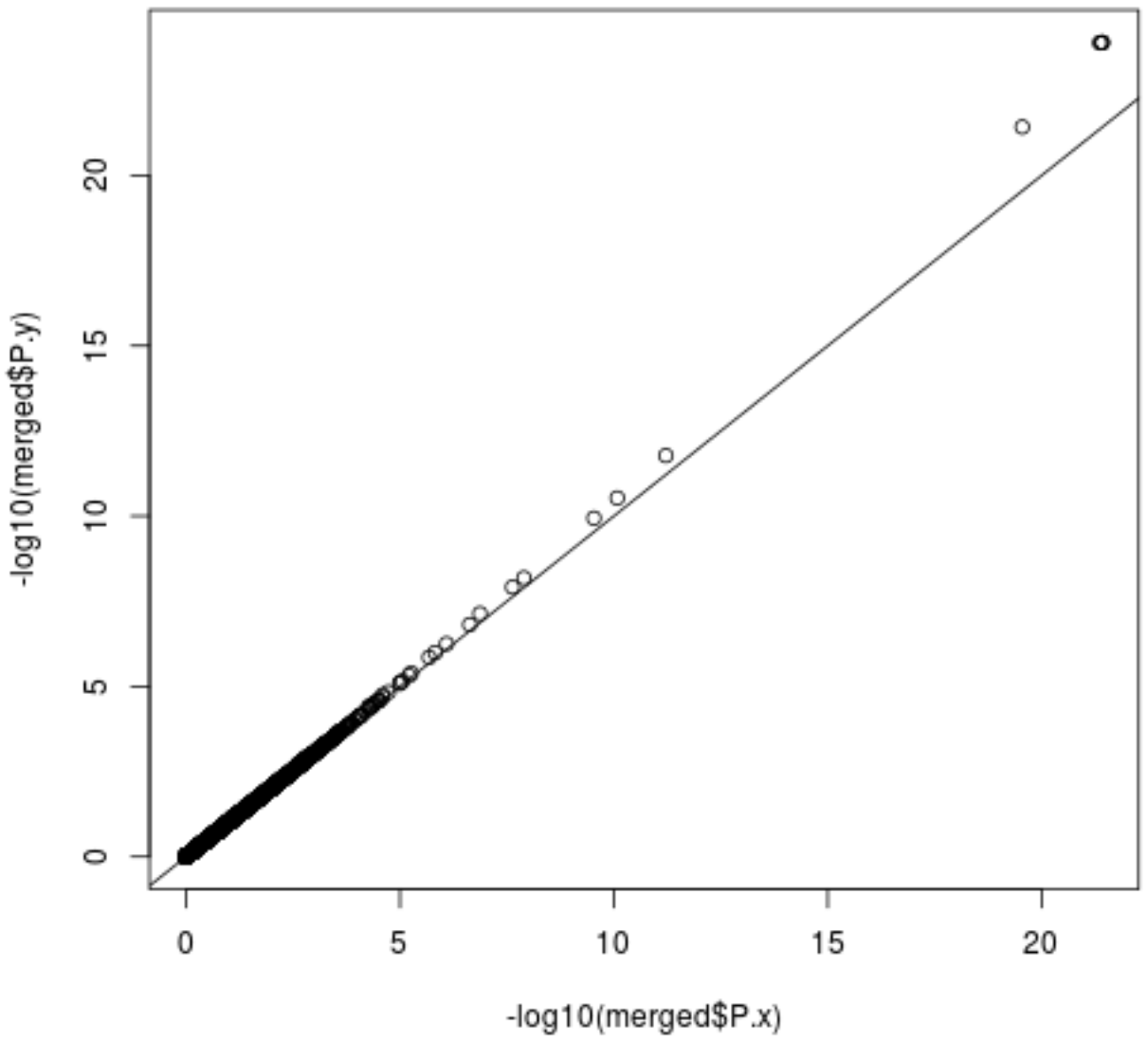

```
png("mh3.png")
manhattan(new3, pch=20, suggestiveline=F, genomewideline=F, ymin=2,
cex.x.axis=0.65, colors=c("black","dodgerblue"), cex=0.5)
dev.off()
```



You should find that the genomic control factor is close to 1.0, and the QQ and Manhattan plots are similar to those you obtained from FaST-LMM.

To compare the results (`res3`) with our previous FaST-LMM results (`res2`), use the following sequence of commands within R:

```
res2<-read.table("FLMMresults", header=T)
new2<-data.frame(res2$SNP, res2$Chromosome, res2$Position, res2$Pvalue)
names(new2)<-c("SNP", "CHR", "BP", "P")
merged=merge(new3,new2, by="SNP", sort=F)

head(res2)
head(new2)
head(new3)
head(merged)

png("compareGCTAFLMM.png")
plot(-log10(merged$P.x),-log10(merged$P.y))
abline(0,1)
dev.off()
```



You should find that the GCTA results (on the x axis) are very similar to the FaST-LMM results (on the y axis), although the -log10 P-values from FaST-LMM are consistently just a little bit higher than those from GCTA.

To use GCTA to estimate the heritability accounted for by all autosomal genome-wide SNPs, you need to first estimate the GRM, and then use the GRM to estimate the (SNP) heritability. This can be achieved using the following commands:

```
gcta64 --bfile quantfamdata --autosome --make-grm-bin --out GCTAgrm
gcta64 --reml --grm-bin GCTAgrm --pheno phenos.txt --out GCTAherit
```

The screen output estimates the SNP heritability V(G)/Vp to be 0.480590 or around 48%.

To estimate the heritabilty accounted for by SNPs on chromosomes 1, 2, 6 and 12 (for example), use the following commands:

```
gcta64 --bfile quantfamdata --chr 1 --make-grm-bin --out GCTAgrmchr1
gcta64 --reml --grm-bin GCTAgrmchr1 --pheno phenos.txt \
--out GCTAheritchr1

gcta64 --bfile quantfamdata --chr 2 --make-grm-bin --out GCTAgrmchr2
gcta64 --reml --grm-bin GCTAgrmchr2 --pheno phenos.txt \
--out GCTAheritchr2

gcta64 --bfile quantfamdata --chr 6 --make-grm-bin --out GCTAgrmchr6
gcta64 --reml --grm-bin GCTAgrmchr6 --pheno phenos.txt \
```

```
--out GCTAheritchr6

gcta64 --bfile quantfamdata --chr 12 --make-grm-bin --out GCTAgrmchr12
gcta64 --reml --grm-bin GCTAgrmchr12 --pheno phenos.txt \
--out GCTAheritchr12
```

You should find that the SNP heritabilities on chromosomes 1 and 2 do not look particularly significant (given the estimated standard errors), but the SNP heritabilities on chromosomes 6 and 12 are significant (as might be expected from the strong effects seen on these chromosomes).

The sum of the SNP heritabilities on these 4 chromosomes (0.206479+0.111512+0.368184+0.286570) adds up to more then the overall SNP heritability of 0.480589. This is due to the phenomenon that, in the presence of population substructure or close relatedness, chromosome-specific heritability estimates can be confounded by shared non-genetic effects (for examples shared environment) or corrrelations between SNPs on different chromosomes, leading to an over-estimate of the chromosome-specific heritability.

To correctly partition the overall heritability between the 22 autosomes, we need to first estimate chromosome-specific GRMs and then include them all simultaneously in the model.

We first calculate the GRMs for all additional chromosomes:

```
gcta64 --bfile quantfamdata --chr 3 --make-grm-bin --out GCTAgrmchr3
gcta64 --bfile quantfamdata --chr 4 --make-grm-bin --out GCTAgrmchr4
gcta64 --bfile quantfamdata --chr 5 --make-grm-bin --out GCTAgrmchr5
gcta64 --bfile quantfamdata --chr 7 --make-grm-bin --out GCTAgrmchr7
gcta64 --bfile quantfamdata --chr 8 --make-grm-bin --out GCTAgrmchr8
gcta64 --bfile quantfamdata --chr 9 --make-grm-bin --out GCTAgrmchr9
gcta64 --bfile quantfamdata --chr 10 --make-grm-bin --out GCTAgrmchr10
gcta64 --bfile quantfamdata --chr 11 --make-grm-bin --out GCTAgrmchr11
gcta64 --bfile quantfamdata --chr 13 --make-grm-bin --out GCTAgrmchr13
gcta64 --bfile quantfamdata --chr 14 --make-grm-bin --out GCTAgrmchr14
gcta64 --bfile quantfamdata --chr 15 --make-grm-bin --out GCTAgrmchr15
gcta64 --bfile quantfamdata --chr 16 --make-grm-bin --out GCTAgrmchr16
gcta64 --bfile quantfamdata --chr 17 --make-grm-bin --out GCTAgrmchr17
gcta64 --bfile quantfamdata --chr 18 --make-grm-bin --out GCTAgrmchr18
gcta64 --bfile quantfamdata --chr 19 --make-grm-bin --out GCTAgrmchr19
gcta64 --bfile quantfamdata --chr 20 --make-grm-bin --out GCTAgrmchr20
gcta64 --bfile quantfamdata --chr 21 --make-grm-bin --out GCTAgrmchr21
gcta64 --bfile quantfamdata --chr 22 --make-grm-bin --out GCTAgrmchr22
```

We then run the analysis:

```
gcta64 --reml --mgrm-bin multipleGRMs.txt --pheno phenos.txt \
--out GCTAherit22GRMs
```

Note this command makes use of a file `multipleGRMs.txt` which we created for you in advance, listing the stem names of the individual GRM files. Unfortunately, in this example the analysis fails to converge, probably because this type of analysis ideally requires a larger number of less closely related individuals.

To instead partition the heritability among two sets of SNPs, chromosome 6

and all other autosomes, we first join together the GRMs for all non-chromosome 6 chromosomes:

```
gcta64 --mgrm-bin multipleGRMsnot6.txt --make-grm --out GCTAgrmnot6
```

Note this command makes use of another file `multipleGRMsnot6.txt` which we created for you in advance, listing the stem names of the individual GRM files (excluding the one for chromosome 6).

We will run the analysis making use of another file `multipleGRMs6andnot6.txt` which we created for you in advance. Take a look at this file and check you understand it.

To run the analysis type:

```
gcta64 --reml --mgrm-bin multipleGRMs6andnot6.txt --pheno phenos.txt \
--out GCTAherit6andnot6
```

The results suggest that a total SNP heritability of 0.469171 can be partitioned as 0.294445 accounted for by chromosome 6, and 0.174726 accounted for by the other autosomes.

3. GCTA fastGWA Analysis

GCTA has an alternative method for performing mixed linear model (MLM)-based GWAS analysis that is partcularly designed for large biobank-scale datasets such as the UK Biobank. Here we will apply it to the (much smaller scale) dataset that we have already analysed.

First we have to make a sparse genetic relationship matrix (GRM) from the full-dense GRM e.g. using a cutoff value of 0.05 (so entries less than 0.05 are set to 0):

```
gcta64 --grm GCTAgrm --make-bK-sparse 0.05 --out GCTAsparsegrm
```

This creates a file GCTAsparsegrm.grm.sp containing the pairs of individuals whose entries in the GRM are greater than 0.05.

For real biobank-scale data, creating the full-dense GRM in the first place can be computationally challenging, and it can be advantageous to partition the GRM into m parts (by row), and compute the parts separately (before joining them back together). To compute i-th part in the current run, we use the syntax `--make-grm-part m i` . For example, use the following commands to re-calculate the full-dense GRM by partitioning the calculation into 3 parts (while also using 5 threads):

```
gcta64 --bfile quantfamdata --make-grm-part 3 1 --thread-num 5 \
--out test
gcta64 --bfile quantfamdata --make-grm-part 3 2 --thread-num 5 \
--out test
gcta64 --bfile quantfamdata --make-grm-part 3 3 --thread-num 5 \
--out test
```

Merge all the parts together:

```
cat test.part_3_*.grm.id > test.grm.id
cat test.part_3_*.grm.bin > test.grm.bin
cat test.part_3_*.grm.N.bin > test.grm.N.bin
```

Now we can create a sparse GRM from this re-calculated version of the full-dense GRM:

```
gcta64 --grm test --make-bK-sparse 0.05 --out newsparsegrm
```

Check that the top few lines of newsparsegrm.grm.sp seem to match GCTAsparsegrm.grm.sp:

```
head *.sp
```

To perform the association analysis, we would normally use the sparse GRM to model close relationships as random effects, while additionally including principal components as fixed effects. So let us first use GCTA to calculate the first 5 principal components:

```
gcta64 --grm-bin GCTAgrm --pca 5 --out pcs
```

To use original GCTAsparsegrm.grm.sp in a fastGWA analysis, type:

```
gcta64 --bfile quantfamdata --grm-sparse GCTAsparsegrm \
--fastGWA-mlm --pheno phenos.txt --qcovar pcs.eigenvec \
--out sparse_assoc
```

To use newsparsegrm.grm.sp in a fastGWA analysis, type:

```
gcta64 --bfile quantfamdata --grm-sparse newsparsegrm \
--fastGWA-mlm --pheno phenos.txt --qcovar pcs.eigenvec \
--out newsparse_assoc
```

Once we have included the principal components, it seems that the estimate of Vg is not statistically significant, and so fastGWA has automatically moved to using linear regression rather than a linear mixed model. This is not really what we wanted! Let us not include principal components - we would then expect the estimate of Vg to be significant, and so in this way we force fastGWA to use a linear mixed model:

```
gcta64 --bfile quantfamdata --grm-sparse GCTAsparsegrm \
--fastGWA-mlm --pheno phenos.txt --out LMMsparse_assoc

gcta64 --bfile quantfamdata --grm-sparse newsparsegrm \
--fastGWA-mlm --pheno phenos.txt --out LMMnewsparse_assoc
```

Let us now use R to check the QQ plots, Manhattan plots and Genomic control factors for these 4 sets of results. Start up R and use the following commands:

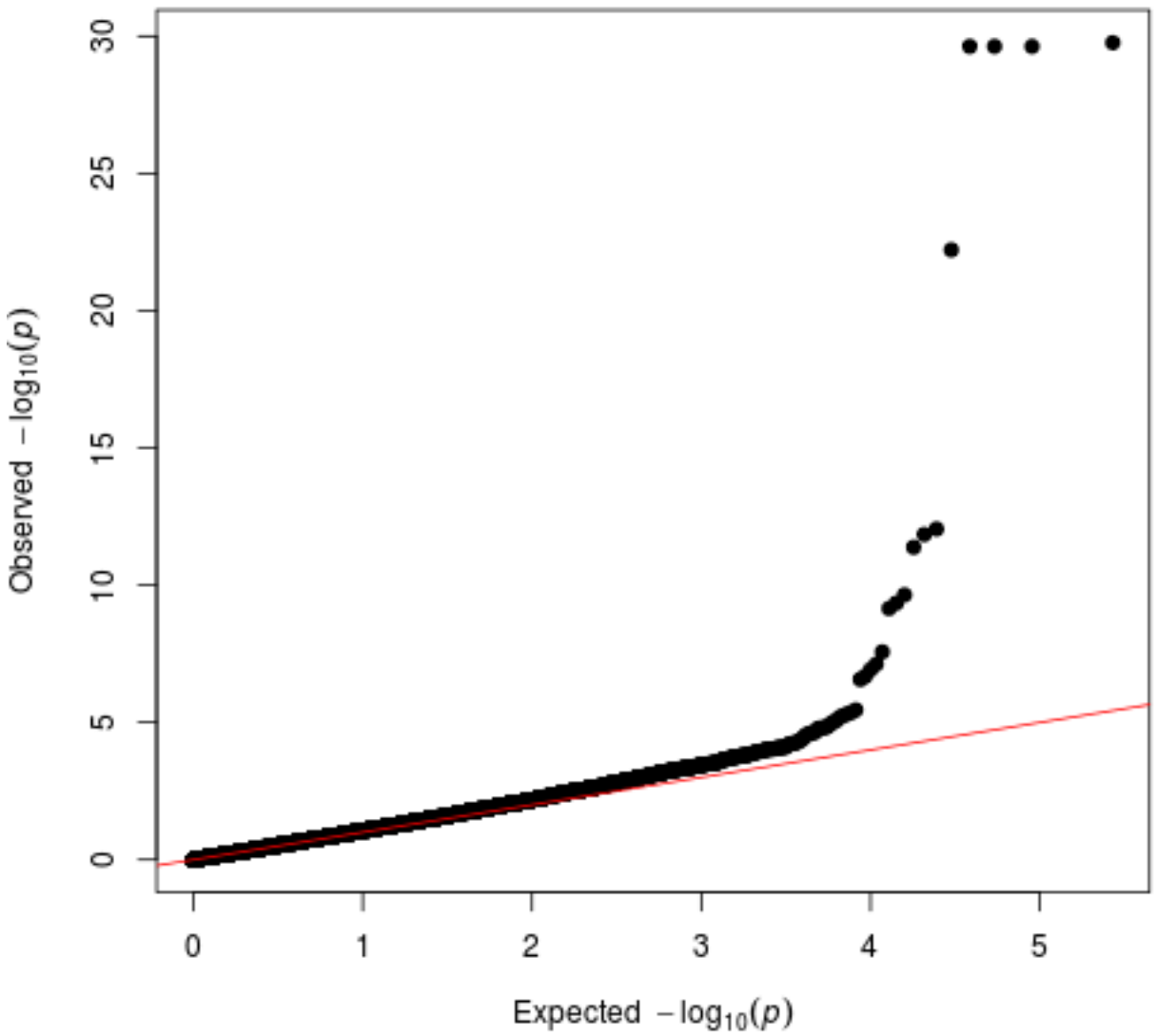
```
res4<-read.table("sparse_assoc.fastGWA", header=T)
res5<-read.table("newsparse_assoc.fastGWA", header=T)
res6<-read.table("LMMsparse_assoc.fastGWA", header=T)
res7<-read.table("LMMnewsparse_assoc.fastGWA", header=T)

head(res4)
head(res5)
head(res6)
```

```
head(res7)

chi<-(qchisq(1-res4$P,1))
lambda=median(chi)/0.456
lambda

chi<-(qchisq(1-res5$P,1))
lambda=median(chi)/0.456
lambda

chi<-(qchisq(1-res6$P,1))
lambda=median(chi)/0.456
lambda

chi<-(qchisq(1-res7$P,1))
lambda=median(chi)/0.456
lambda
```

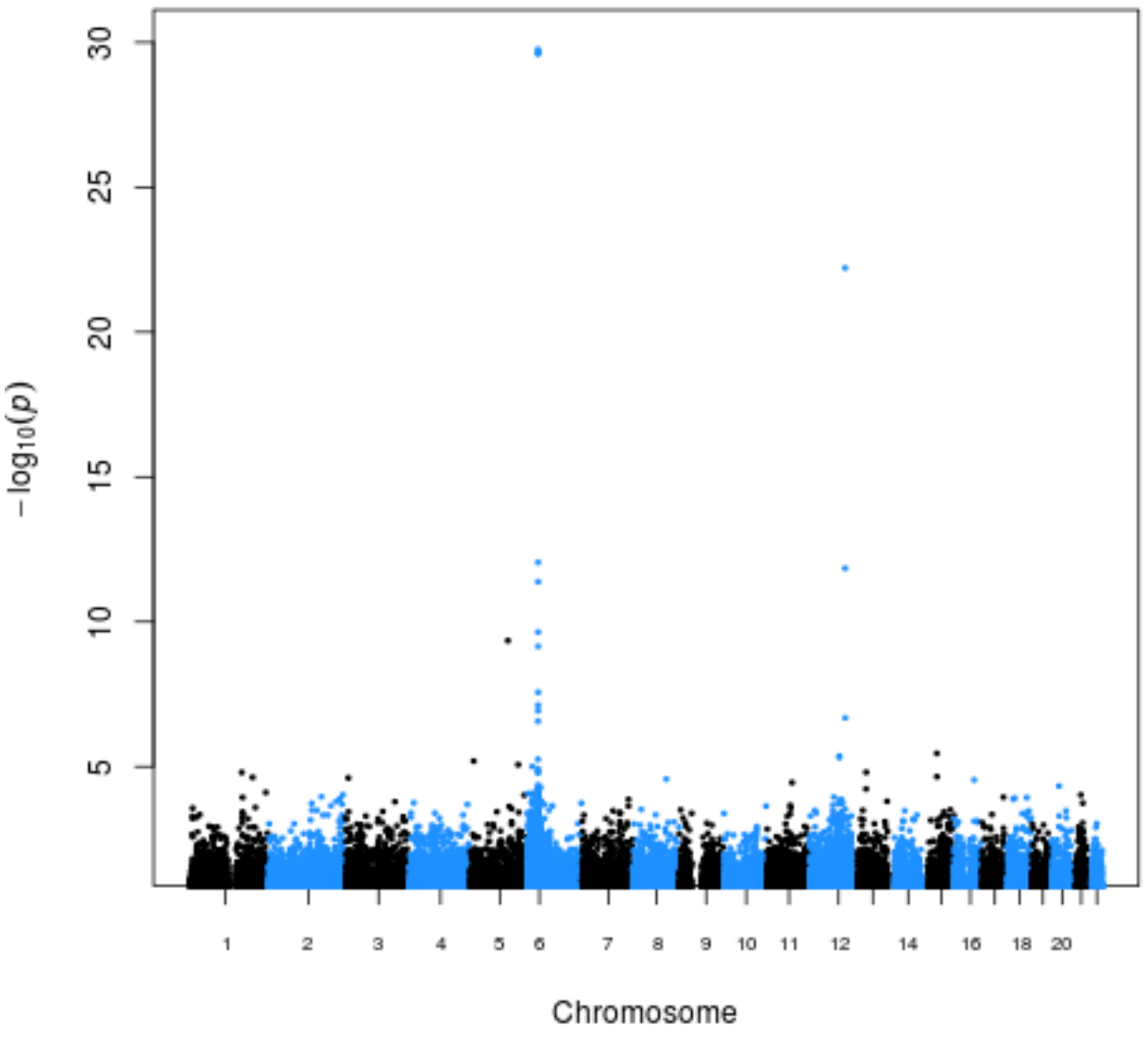
You should see that the linear mixed model (with a sparse GRM) does a better job at controlling for relatedness (giving lambda closer to 1.0) than just including 5 principal components.

Now continue in R to make the plots for these 4 analyses (which will be very similar to those you made previously):

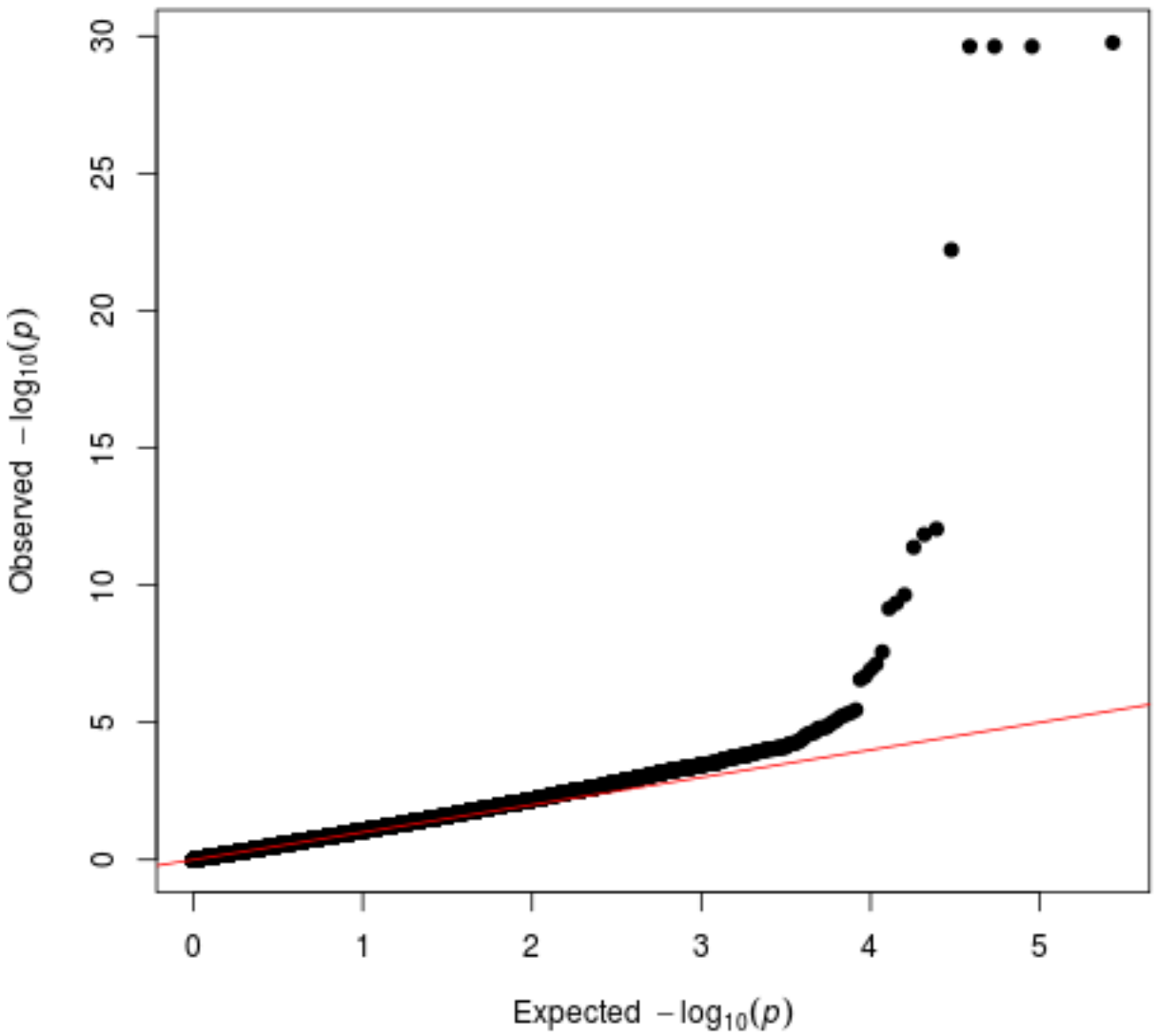
```
source("qqmanHJCupdated.R")

new4<-data.frame(res4$SNP, res4$CHR, res4$POS, res4$P)
names(new4)<-c("SNP", "CHR", "BP", "P")

png("qq4.png")
qq(new4$P)
dev.off()
```



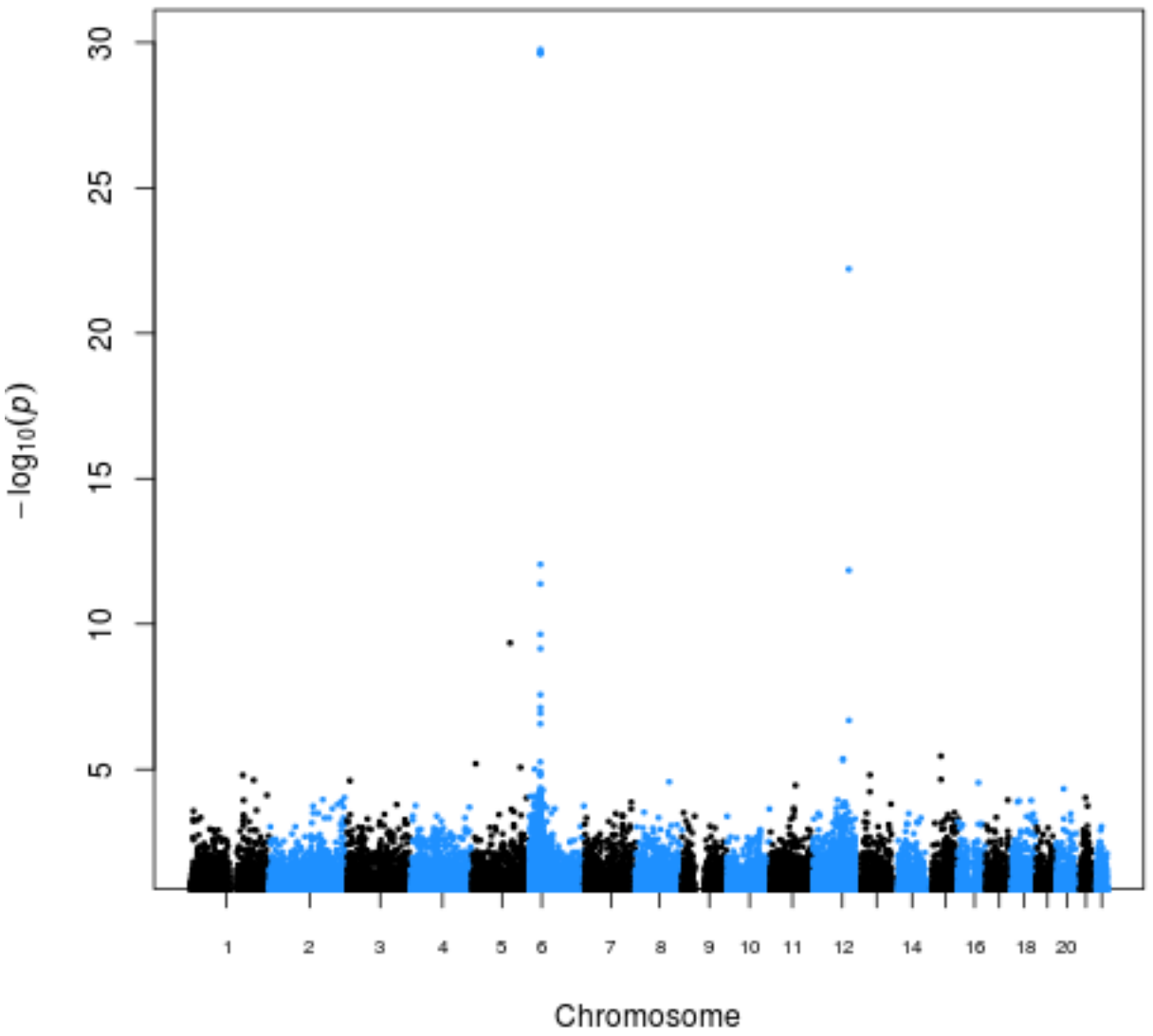

```
png("mh4.png")
manhattan(new4, pch=20, suggestiveline=F, genomewideline=F, ymin=2,
cex.x.axis=0.65, colors=c("black","dodgerblue"), cex=0.5)
dev.off()
```



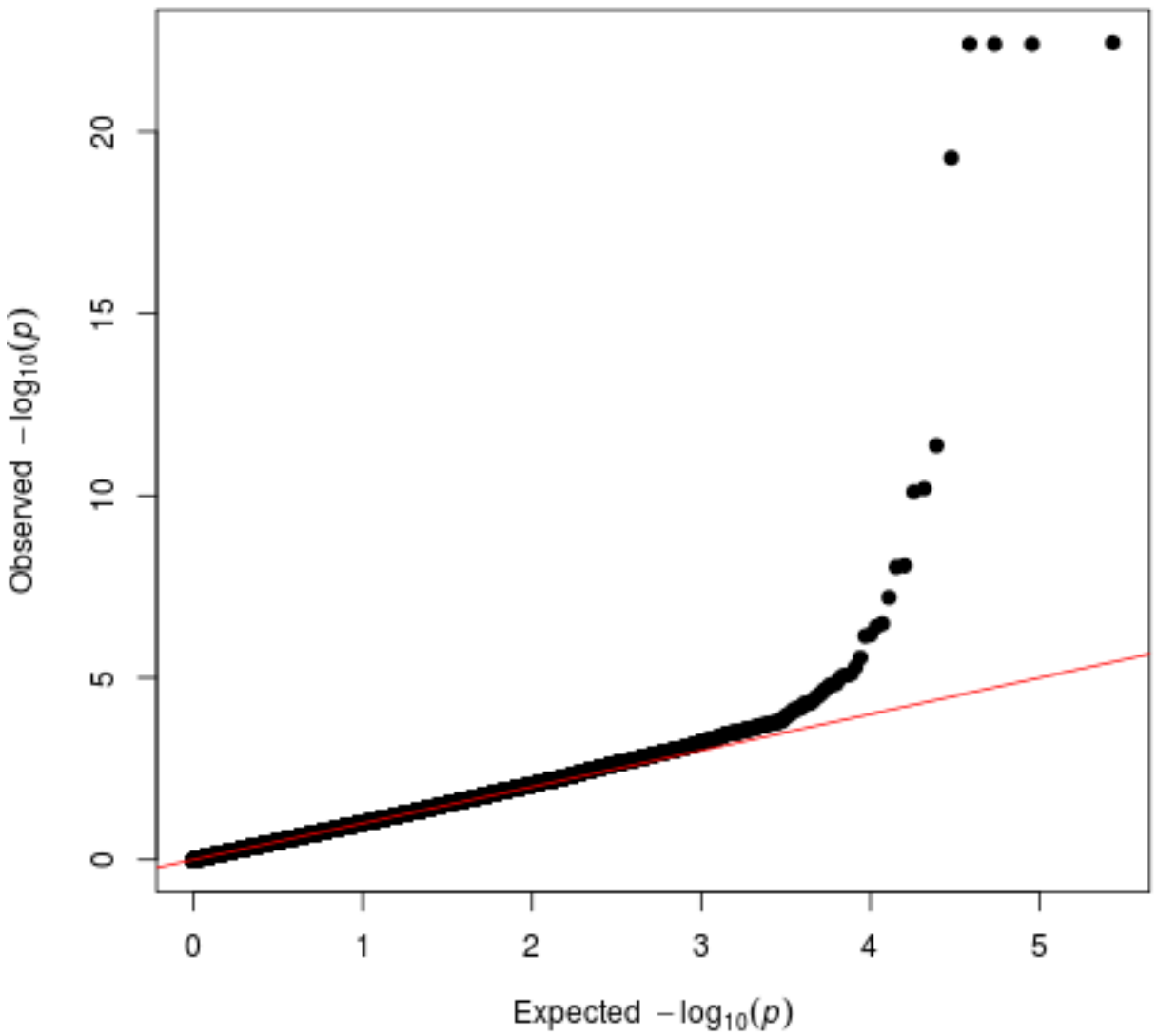

```
new5<-data.frame(res5$SNP, res5$CHR, res5$POS, res5$P)
names(new5)<-c("SNP", "CHR", "BP", "P")

png("qq5.png")
qq(new5$P)
dev.off()
```



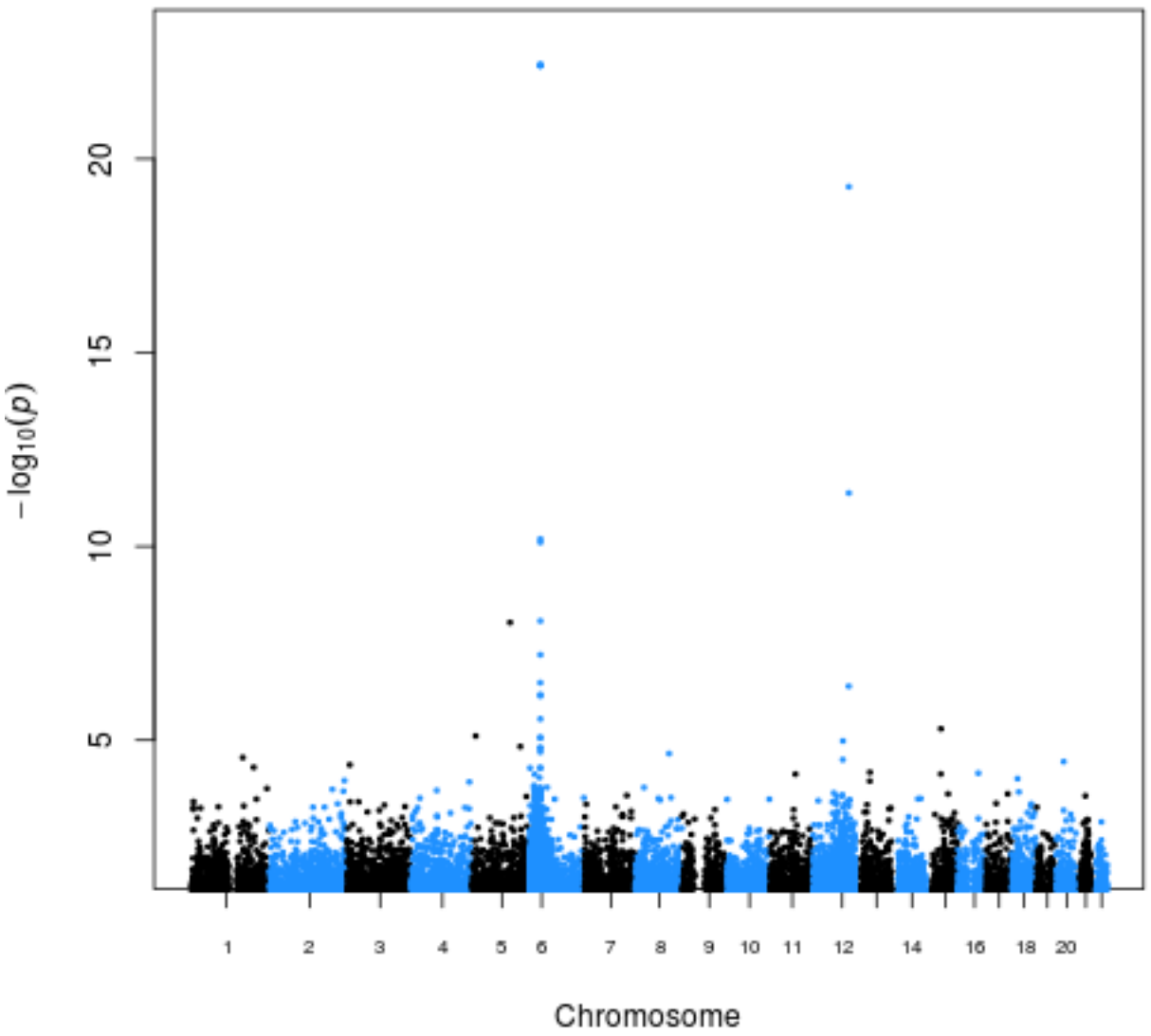

```
png("mh5.png")
manhattan(new5, pch=20, suggestiveline=F, genomewideline=F, ymin=2,
cex.x.axis=0.65, colors=c("black","dodgerblue"), cex=0.5)
dev.off()
```



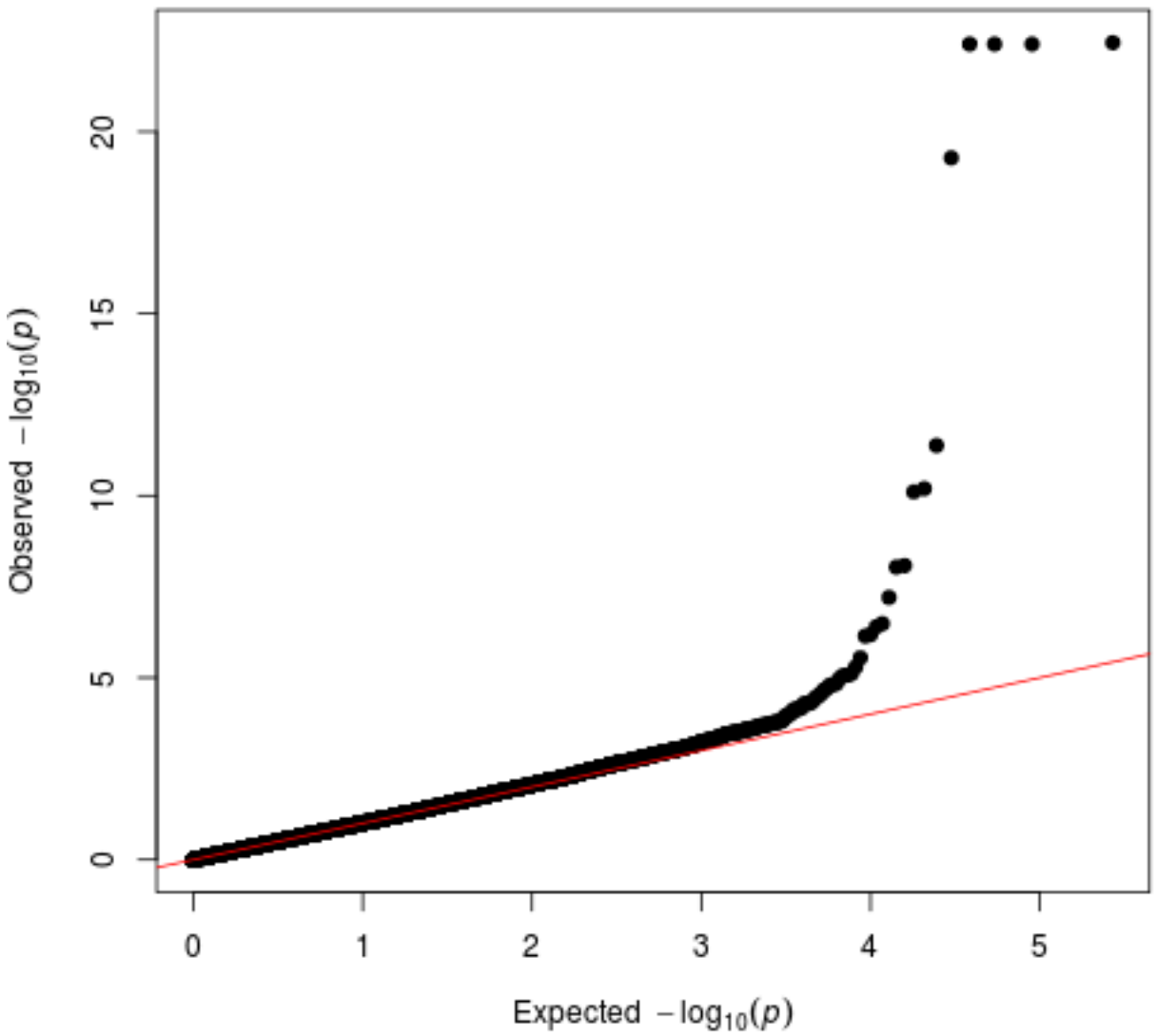

```
new6<-data.frame(res6$SNP, res6$CHR, res6$POS, res6$P)
names(new6)<-c("SNP", "CHR", "BP", "P")

png("qq6.png")
qq(new6$P)
dev.off()
```



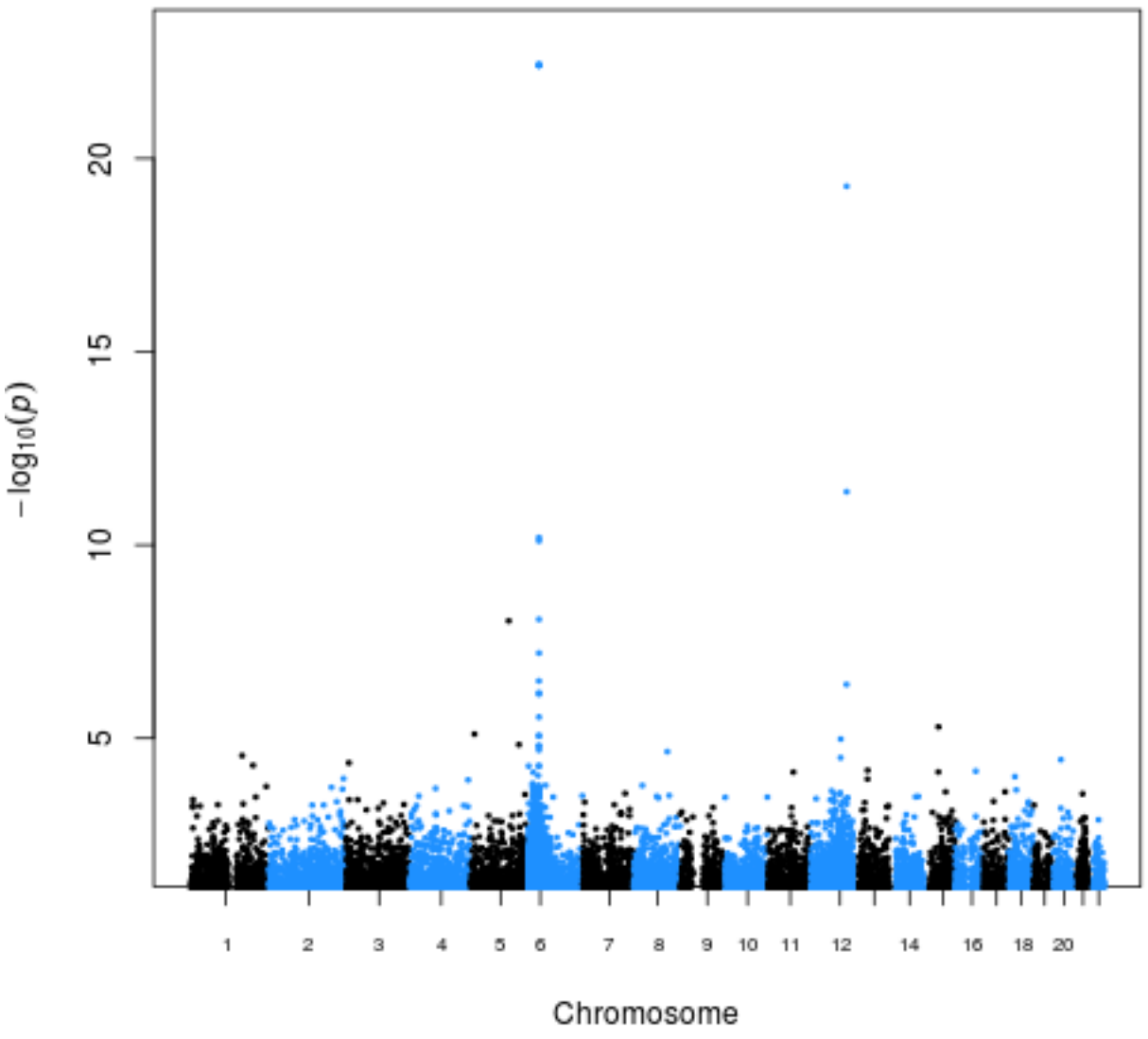

```
png("mh6.png")
manhattan(new6, pch=20, suggestiveline=F, genomewideline=F, ymin=2,
cex.x.axis=0.65, colors=c("black","dodgerblue"), cex=0.5)
dev.off()
```




```
new7<-data.frame(res7$SNP, res7$CHR, res7$POS, res7$P)
names(new7)<-c("SNP", "CHR", "BP", "P")

png("qq7.png")
qq(new7$P)
dev.off()
```

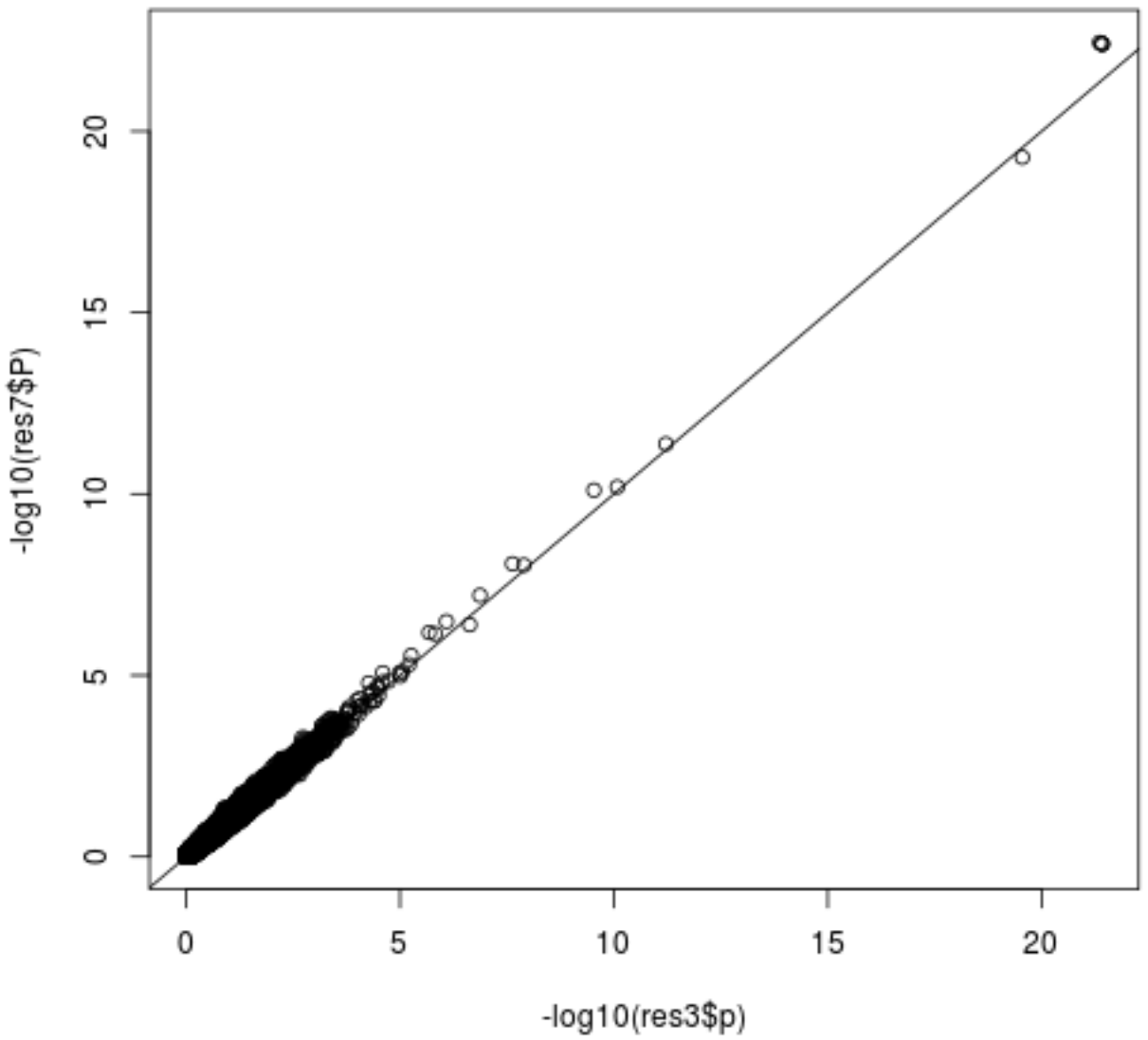



```
png("mh7.png")
manhattan(new7, pch=20, suggestiveline=F, genomewideline=F, ymin=2,
cex.x.axis=0.65, colors=c("black","dodgerblue"), cex=0.5)
dev.off()
```



Finally we will check how the results from the linear mixed model (with a sparse GRM) from fastGWA compares to the original GCTA `--mlma` results:

```
res3<-read.table("GCTAresults.mlma", header=T)
res7<-read.table("LMMnewsparse_assoc.fastGWA", header=T)
head(res3)
head(res7)

png("compareGCTAfastGWA.png")
plot(-log10(res3$p),-log10(res7$P))
abline(0,1)
dev.off()
```



As you can see, the results are extremely similar.

Answers

How to interpret the output

Interpretation of the output is described in the step-by-step instructions. Please ask if you need help in understanding the output.

Comments

Other packages

Another package that can implement a similar analysis to GCTA is DISSECT

References

Yang et al. (2011) GCTA: A tool for genome-wide complex trait analysis. American Journal of Human Genetics, 88:76-82.

Exercises prepared by: Heather Cordell
Checked by:
Programs used: R, GCTA
Last updated: 01/15/2021 14:53:5501/15/2021 14:53:36