Module 18: Assorted Practical

Need help? doug.speed@ucl.ac.uk

Check List

If all has gone to plan you should:

- Have `putty` installed and configured, which allows you to "ssh" into Hong's server, hong1.une.edu.au (username asc2016, pw feb01). This is where you will run `PLINK, GCTA, LDAK and IMPUTE2`
- Have `winSCP` installed and configured, which allows you to copy files from Hong's server to your Desktop (which you can then read into R), and files from your Desktop to Hong's server - NOT AS IMPORTANT IN THIS ONE
- Have a copy of the slides, either from Julius' website `http://jvanderw.une.edu.au/AGSCcourse.htm`, or from the module18 folder on Hong's server

If so, you may now begin :)

1 - Imputation

We will now perform phasing and imputation using SHAPEIT and IMPUTE2. For this, we require:

Genotype data: genofinal.bed, genofinal.bim, genofinal.fam

These data should be thoroughly QC'ed (both individuals and SNPs, and population outliers excluded / treated separately

Recombination map / genetic distances (for phasing and imputation)

genetic_map_chr#_combined_b37.short (one # for each chr)

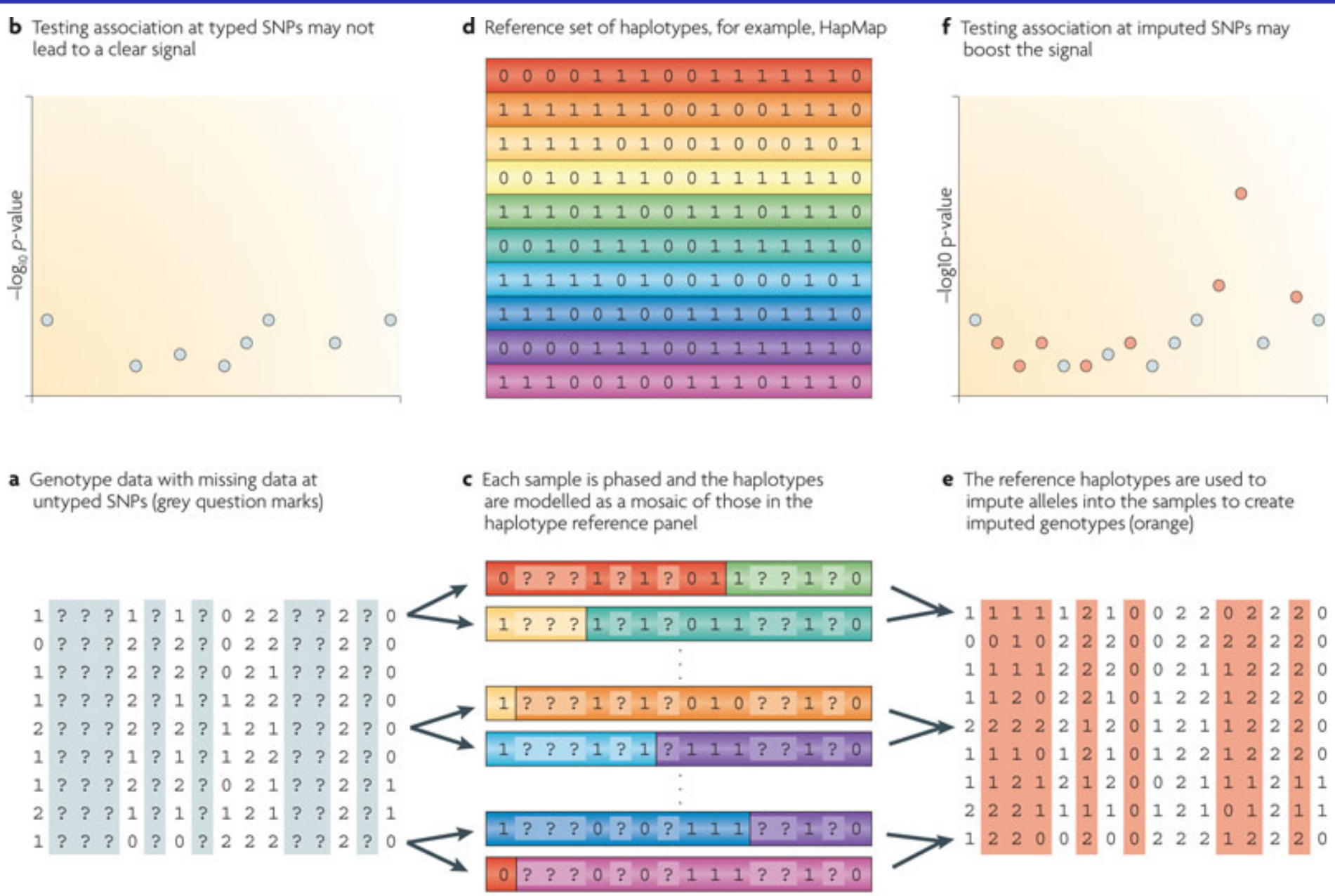
Reference haplotypes (for imputation)

ALL_1000G_phase1integrated_v3_chr#_short.hap.gz

ALL_1000G_phase1integrated_v3_chr#_short.legend.gz

```
cd $HOME/doug
cp ../module18/* ./
```

1 - Imputation



1 - Imputation

First we divide the data by chromosome

```
../plink --bfile genofinal --chr 1 --make-bed --out split1

../plink --bfile genofinal --chr 2 --make-bed --out split2

../plink --bfile genofinal --chr 23 --make-bed --out split23
```

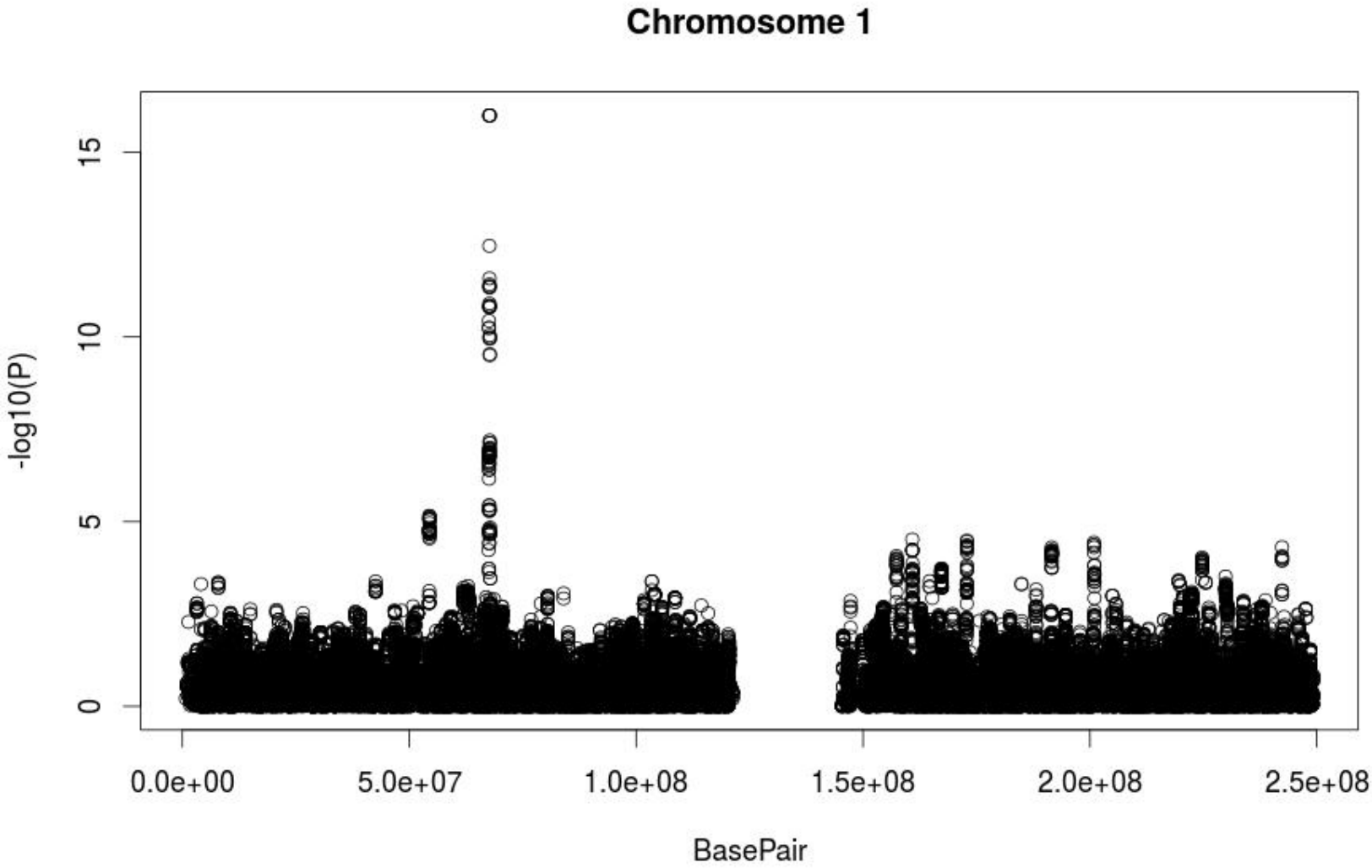
1 - Imputation

Then we phase the data (each chromosome separately)

```
../shapeit -B split1 -M genetic_map_chr1_combined_b37.short --thread 4 \
--effective-size 11418 -O chr1

../shapeit -B split2 -M genetic_map_chr2_combined_b37.short --thread 4 \
--effective-size 11418 -O chr2

../shapeit -B split23 -M genetic_map_chrX_nonPAR_combined_b37.short --thread 4 \
--effective-size 11418 -O chr23
```

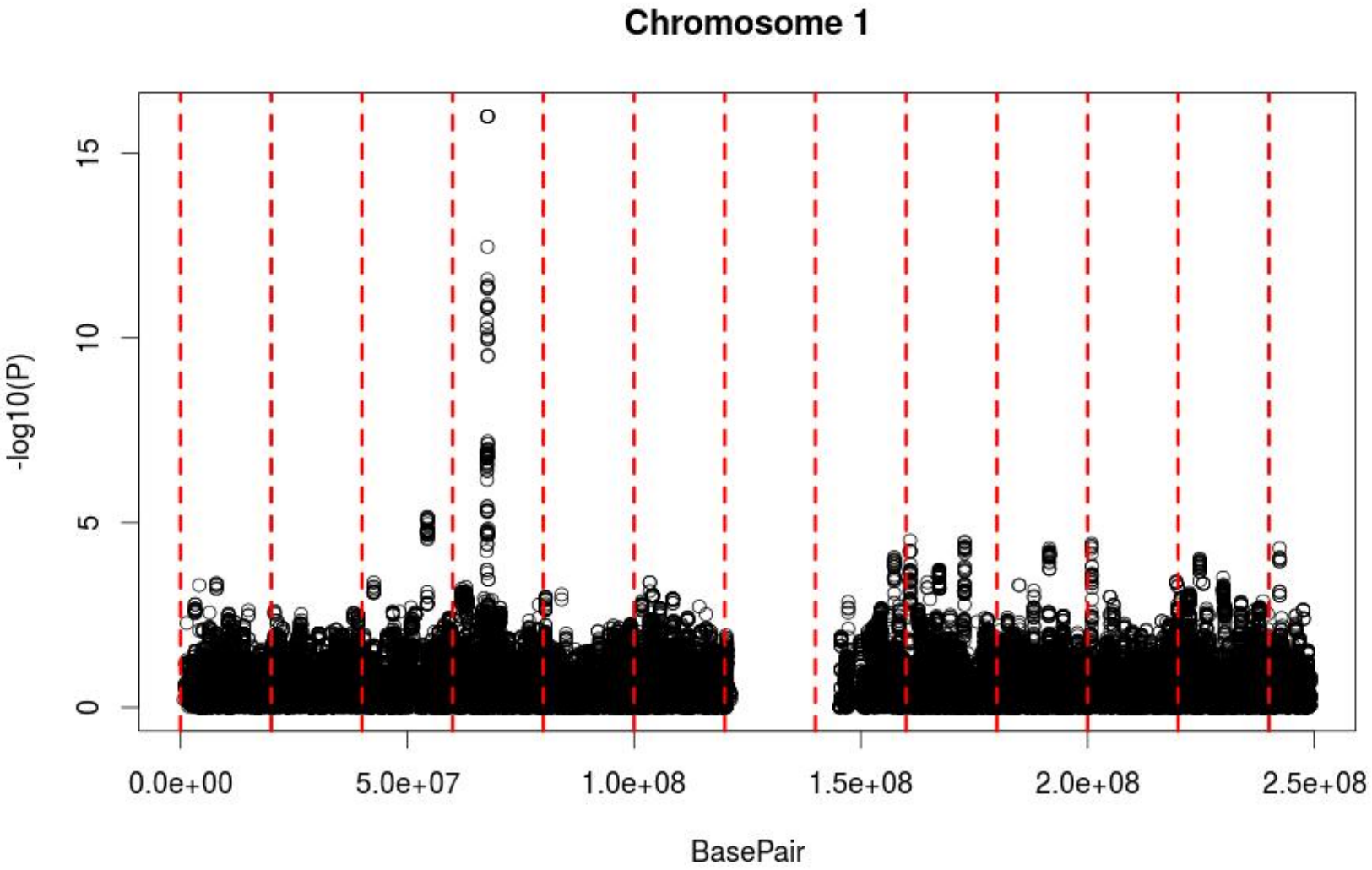
1 - Imputation

We impute the data in chunks (here, about 5 Mb)



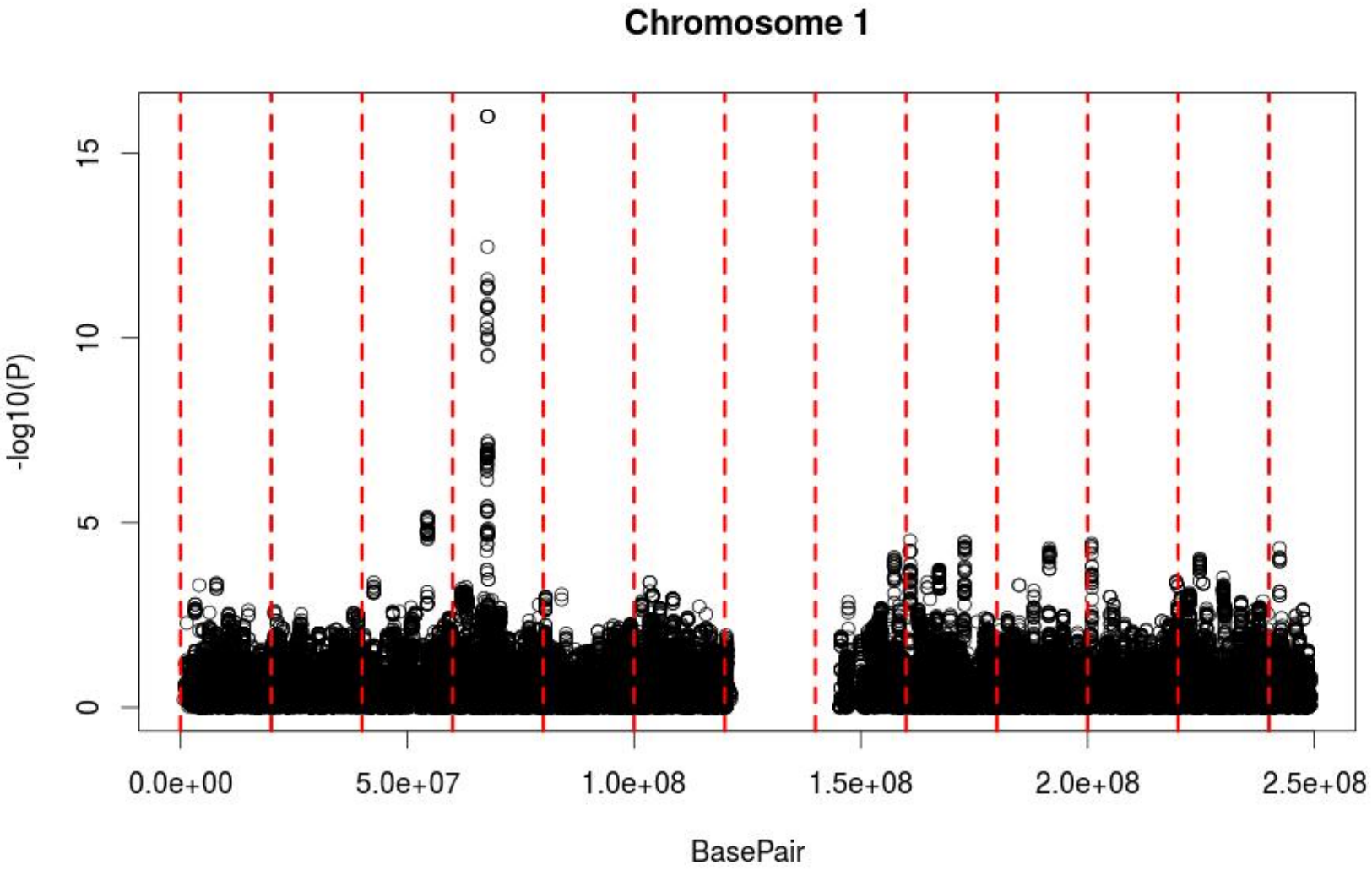
1 - Imputation

The genome should be "intelligently" divided



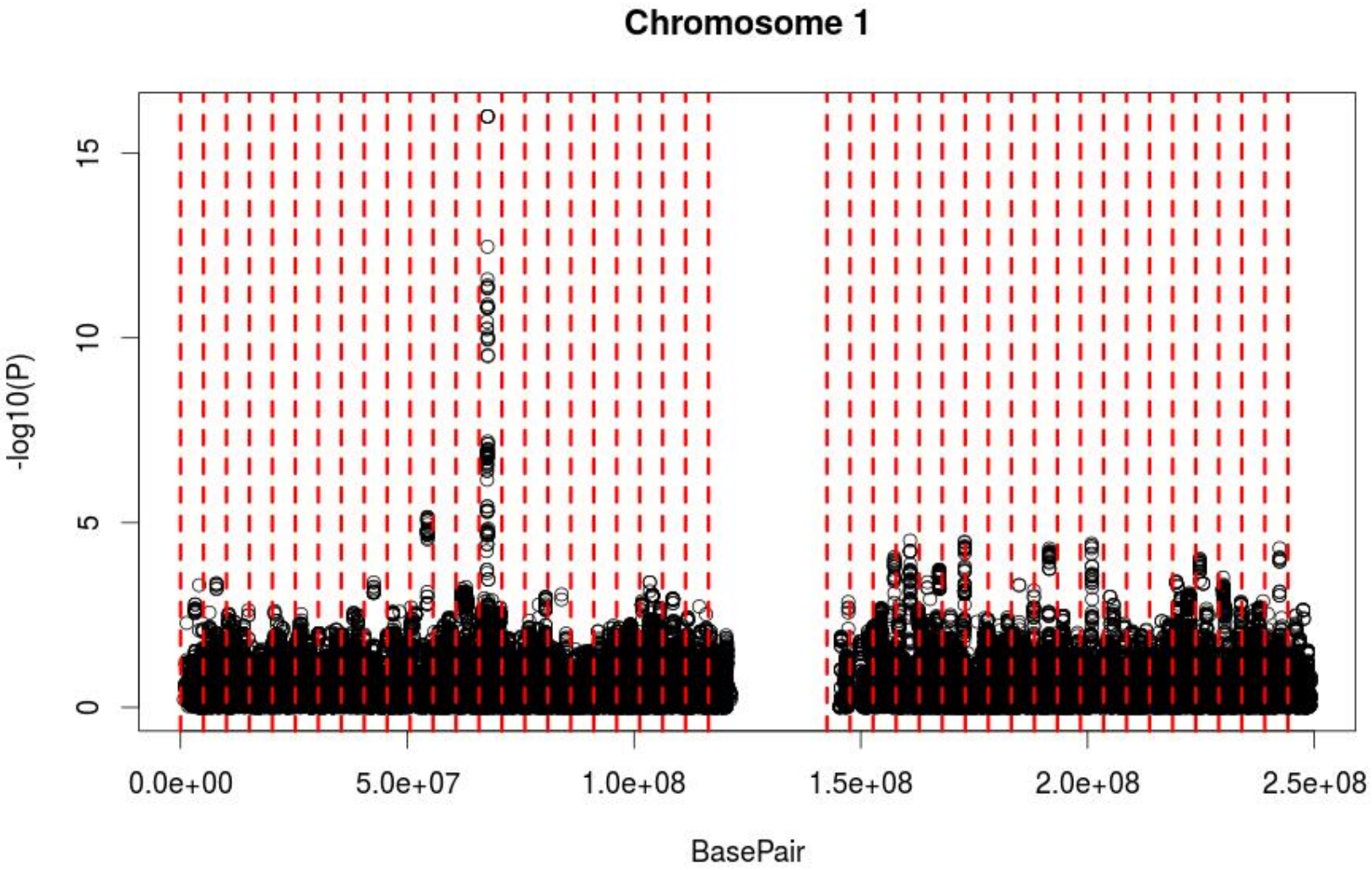
1 - Imputation

The genome should be "intelligently" divided



1 - Imputation

The genome should be "intelligently" divided



1 - Imputation

We impute the data in chunks (here, about 5 Mb)

```
$ head allreg.txt
1 10469 5071806 69370
1 5071807 10133144 70807
1 10133145 15194482 65773
1 15194483 20255820 74979
1 20255821 25317158 64635
1 25317159 30378496 57341
1 30378497 35439834 67053
1 35439835 40501172 63018
1 40501173 45562510 62898
1 45562511 50623848 59120
```

1 - Imputation

We impute the data in chunks (here, about 5 Mb)

```
for number in {1,2,3,46,47};
do
chr=‘sed -n ${number}p allreg.txt | awk ’{print $1}’‘
start=‘sed -n ${number}p allreg.txt | awk ’{print $2}’‘
end=‘sed -n ${number}p allreg.txt | awk ’{print $3}’‘

echo "
../impute2 \\
   -m genetic_map_chr${chr}_combined_b37.short \\
   -h ALL_1000G_phase1integrated_v3_chr${chr}_short.hap.gz \\
   -l ALL_1000G_phase1integrated_v3_chr${chr}_short.legend.gz \\
   -use_prephased_g \\
   -known_haps_g chr${chr}.haps \\
   -int $start $end \\
   -Ne 11418 \\
   -o_gz -o chunk$number \\
   -allow_large_regions -seed 36946
" > script$number
done
```

1 - Imputation

```
chr=‘sed -n ${number}p allreg.txt | awk ’{print $1}’‘
start=‘sed -n ${number}p allreg.txt | awk ’{print $2}’‘
end=‘sed -n ${number}p allreg.txt | awk ’{print $3}’‘
```

These are asking bash to read Row “number” of file `allreg.txt` and extract elements 1 (chromosome), 2 (start of chunk) and 3 (end of chunk)

```
> chr=`sed -n ${number}p allreg.txt | awk '{print $1}'`
> start=`sed -n ${number}p allreg.txt | awk '{print $2}'`
> end=`sed -n ${number}p allreg.txt | awk '{print $3}'`
>
```

Note, they are back-ticks at start and end, then single quotation marks around the curly brackets!!!!!!!

1 - Imputation

Then we construct the imputation command (one for each chunk)

`-m`, `-h`, `-l` are telling it where to look for the recombination map and reference data

We add `-use_prephased_g` and `-known_haps_g` to tell IMPUTE2 we have already performed the phasing, and where these are stored

`-int $start $end` defines the chunk we are imputing

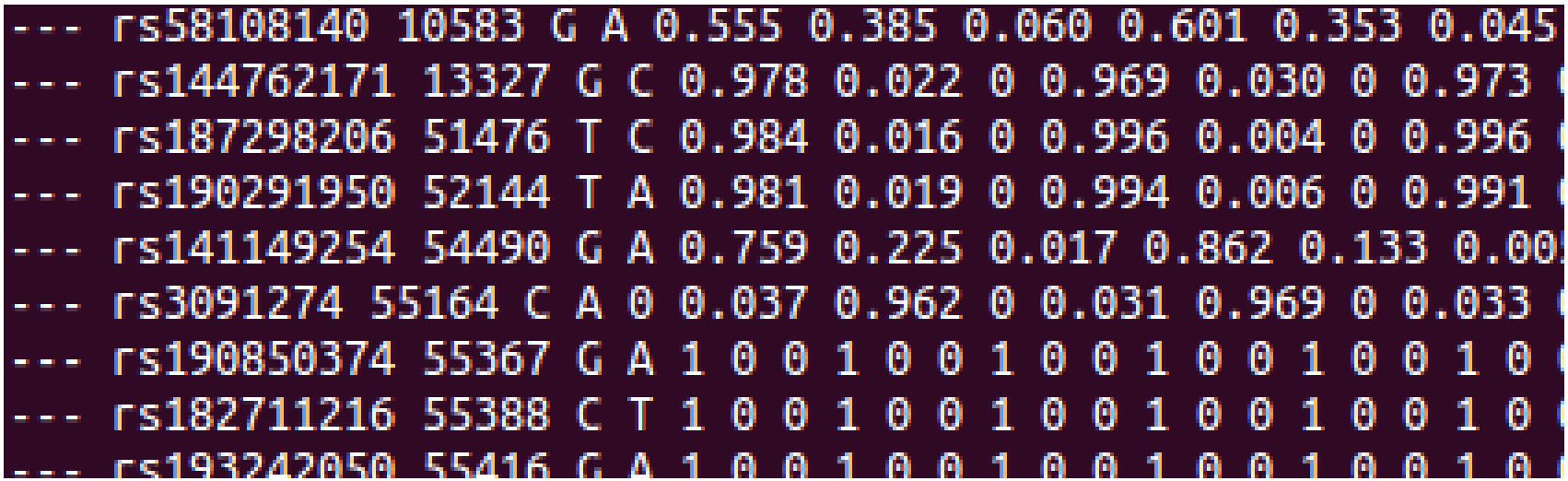
`-Ne` is an estimate of the effective population size

Could add `-k XXX` (by default 500) to tell IMPUTE2 how many reference haplotypes to use when imputing each individual
https://mathgen.stats.ox.ac.uk/impute/impute_v2.html

1 - Imputation

```
$ ls chunk1*
chunk1.gz  chunk1_info  chunk1_info_by_sample
chunk1_summary  chunk1_warnings
```

chunk1.gz



Each row is a SNP, with five headers, then three probabilities for each individual: $\mathbb{P}$(SNP=A1A1), $\mathbb{P}$(SNP=A1A2), $\mathbb{P}$(SNP=A2A2)

1 - Imputation

```
$ ls chunk1*
chunk1.gz  chunk1_info  chunk1_info_by_sample
chunk1_summary  chunk1_warnings
```

chunk1_info

```
snp_id rs_id position a0 a1 exp_freq_a1 info certainty type info_type0 concord_type0 r2_type0
--- rs58108140 10583 G A 0.207 0.202 0.694 0 -1 -1 -1
--- rs144762171 13327 G C 0.043 0.166 0.922 0 -1 -1 -1
--- rs187298206 51476 T C 0.010 0.318 0.983 0 -1 -1 -1
--- rs190291950 52144 T A 0.017 0.232 0.969 0 -1 -1 -1
--- rs141149254 54490 G A 0.151 0.191 0.760 0 -1 -1 -1
--- rs3091274 55164 C A 0.952 0.235 0.918 0 -1 -1 -1
--- rs190850374 55367 G A 0.000 0.000 1.000 0 -1 -1 -1
--- rs182711216 55388 C T 0.000 0.000 1.000 0 -1 -1 -1
--- rs193242050 55416 G A 0.000 0.000 1.000 0 -1 -1 -1
```

Gives measures of quality for each SNP: (expected) MAF, INFO Score, (expected) Callrate, and others

Most important are Columns 7 and 10, which provide INFOs for imputed SNPs then genotyped SNPs

1 - Imputation

The genotypes ("dosage data") are stored in Chiamo — Oxford Stat Format, which can then be provided as options to `PLINK` and `LDAK` (or `SNPTEST`)

https://www.cog-genomics.org/plink2/formats#gen

http://dougspeed.com/file-formats/

The INFOs for genotyped SNPs (Column 10) are incredibly powerful for detecting genotyping errors (they are obtained by leaving that SNP out and seeing how well it's predicted values match those observed), so I use IMPUTE2 for QC even if not using the imputed data

Mice Data

Genotypes stored in mice.bed, mice.bim and mice.fam

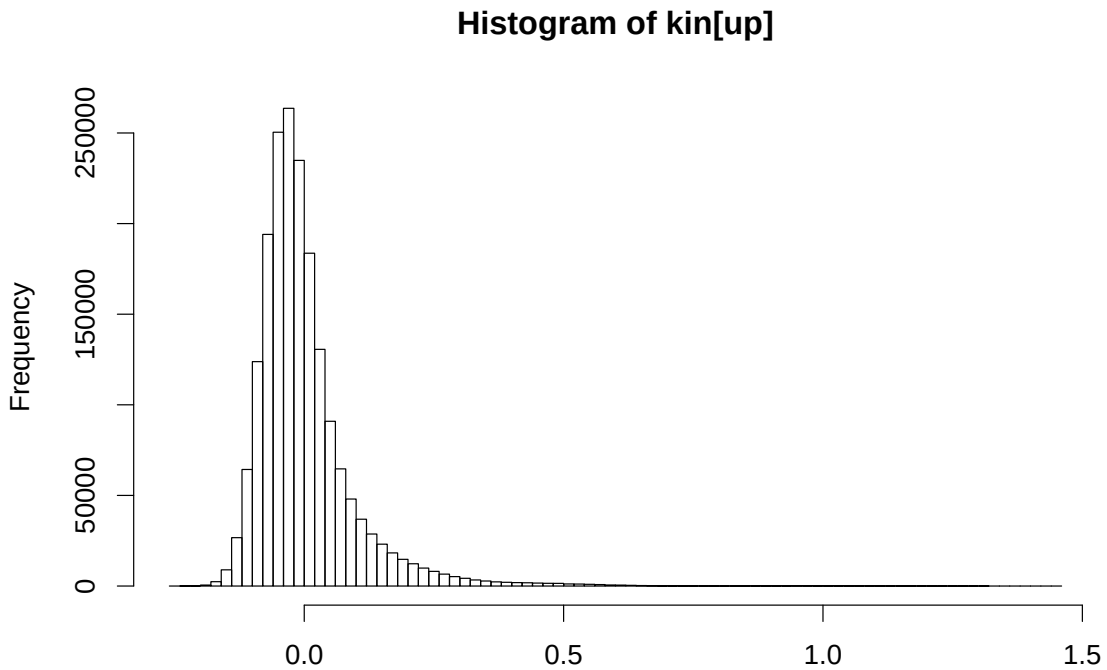
1940 mice genotyped for 8516 SNP

`mice.pheno` contains 143 phenotypes

2a - Heritability Analysis

```
../ldak.mm --calc-kins-direct micekins --bfile mice \
--ignore-weights YES --kinship-raw YES
```

```
> kin=as.matrix(read.table("micekins.grm.raw"))
> up=which(upper.tri(kin,diag=T))
> hist(kin[up],n=100,xlab="Estimated Kinship")
```



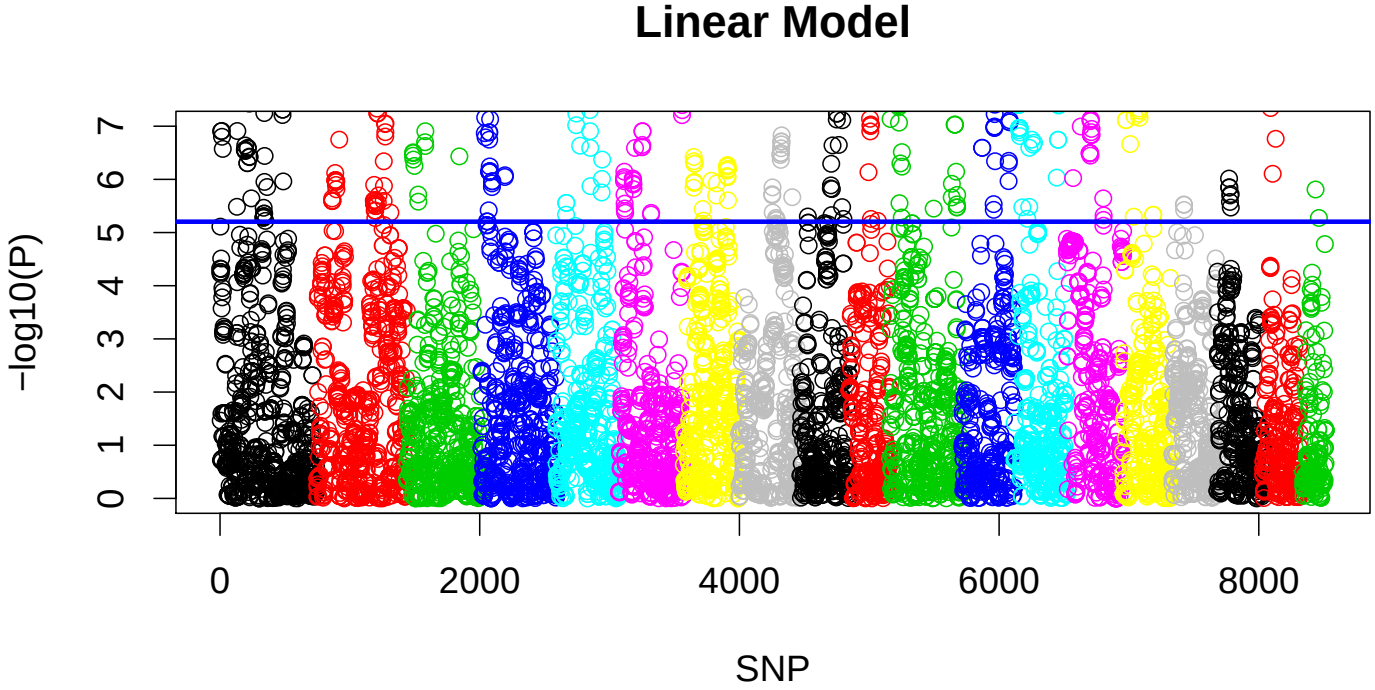
2a - Heritability Analysis

```
../ldak.mm --reml mice --pheno mice.pheno --mpheno 1 \
--grm micekins --eigen-save micekins
../ldak.mm --reml mice --pheno mice.pheno --mpheno 1 \
--grm micekins --eigen micekins

cat mice.reml
Total_Samples 1845
With_Phenotypes 1845
LRT_Stat 212.8907
LRT_P 1.0000e-16
Component Heritability Her_SD Size Intensity Int_SD
Her_K1 0.223403 0.031165 8516.00 2.623333 0.365958
Her_All 0.223403 0.031165 8516.00 2.623333 0.365958
```

2b - Linear Regression

```
../ldak.mm --linear mice --pheno mice.pheno --mpheno 1 \
--bfile mice
```

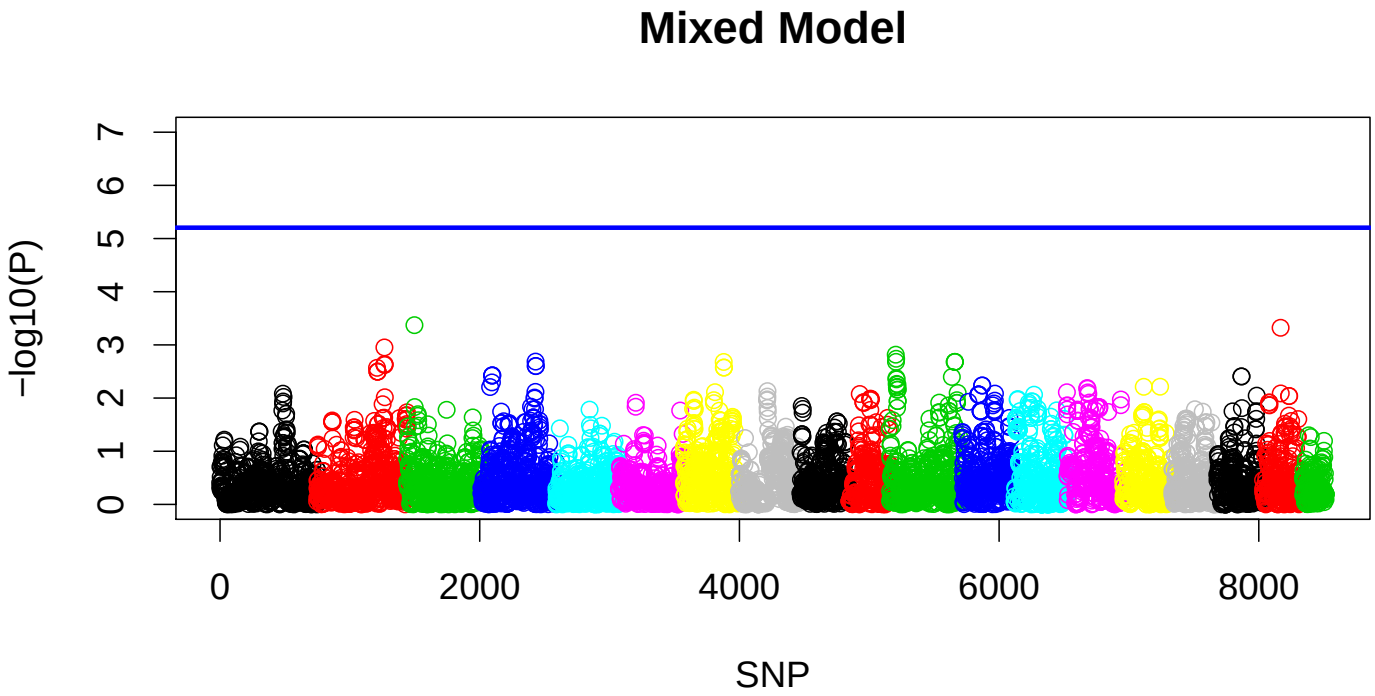
```
> res=as.matrix(read.table("mice.assoc",head=T))
> chr=as.numeric(res[,1]);pva=as.numeric(res[,11])
> plot(-log10(pva),col=chr,xlab="SNP",ylab="-log10(P)")
```



2b - Mixed Model Linear Regression

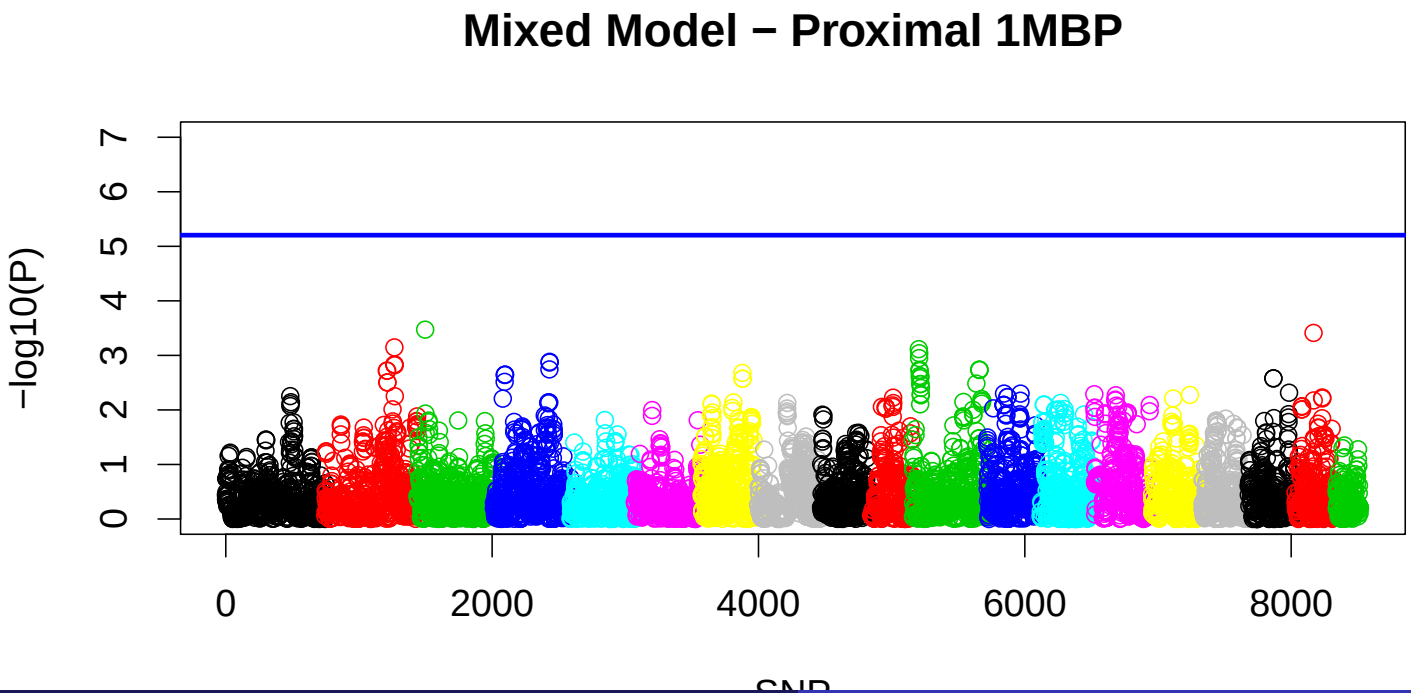
```
../ldak.mm --linear micemm --pheno mice.pheno --mpheno 1 \
--bfile mice --grm micekins --eigen micekins
```

```
> res2=as.matrix(read.table("micemm.assoc",head=T))
> pva2=as.numeric(res2[,11])
> plot(-log10(pva2),col=chr,xlab="SNP",ylim=c(0,7))
```



2c - Proximal Contamination - Sorry, Complicated

```
../ldak.mm --linear micemmp --pheno mice.pheno --grm micekins \
--mpheno 1 --bfile mice  --proximal YES --proximal-buffer 1000000 \
--eigen micekins --projection-save mice

../ldak.mm --linear micemmp --pheno mice.pheno --grm micekins \
--mpheno 1 --bfile mice  --proximal YES --proximal-buffer 1000000 \
--eigen micekins --projection mice
```



3 - Gene-Based Analysis

Cut into genes (chunks) - three possible ways

```
#divide into chunks of fixed bp
../ldak.out --cut-genes chunks --chunks-bp 1000000 \
--bfile mice --ignore-weights YES

#divide into chunks of fixed number of SNPs
#../ldak.out --cut-genes chunks --chunks 100 \
--bfile mice --ignore-weights YES

#divide into genes based on a gene file
#../ldak.out --cut-genes chunks --genefile genes.txt \
--bfile mice --ignore-weights YES
```

3 - Gene-Based Analysis

Test the genes (chunks) for association (one partition at a time)

```
../ldak.out --calc-genes-reml chunks --pheno mice.pheno \
--bfile mice --partition 1 --ignore-weights YES  --mpheno 1
```

Join the results across partitions

```
../ldak.out --join-genes-reml chunks
```

```
ls chunks
effects1  gene_details.txt  gene_tallies.txt  regress1
gammas    gene_preds.txt    maximums          regressALL
```

3 - Gene-Based Analysis

The main results file is `chunks/regressALL`

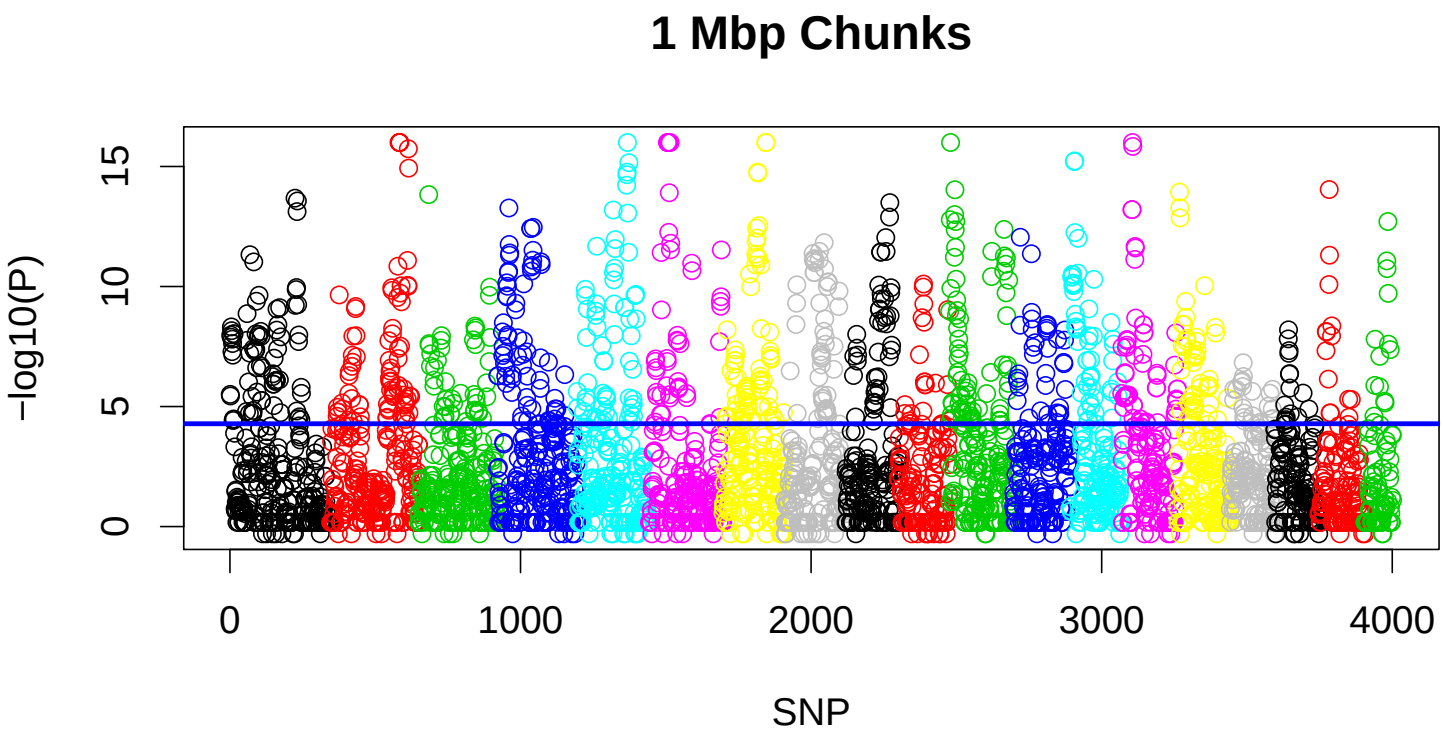
```
doug@doug-laptop:~/Dropbox/armidale2/Mod_18_Imp_Pract/practical5$ head chunks/regressALL
Gene_Number Gene_Name Phen_Number REML_Her REML_SD Alt_Like Null_Like LRT_Stat LRT_P_Raw LRT_P_Perm BLANK Score_Stat
ore_P BLANK Gene_Length Gene_Weight
1 Chunk_1 1 0.008611 0.010618 1608.9052 1601.4856 14.8393 5.8535e-05 3.5531e-05 -99 2.0835 1.8602e-02 -99 2 3.00
2 Chunk_2 1 0.009648 0.012535 1608.7979 1601.4856 14.6247 6.5593e-05 3.9659e-05 -99 2.0539 1.9992e-02 -99 2 4.00
3 Chunk_3 1 0.040061 0.041809 1612.9139 1601.4856 22.8567 8.7271e-07 6.1902e-07 -99 2.5907 4.7896e-03 -99 3 3.00
4 Chunk_4 1 0.028357 0.039441 1612.7655 1601.4856 22.5598 1.0185e-06 7.1811e-07 -99 2.4327 7.4934e-03 -99 4 6.00
5 Chunk_5 1 0.016741 0.015699 1613.4623 1601.4856 23.9536 4.9343e-07 3.5793e-07 -99 2.5592 5.2460e-03 -99 3 3.00
6 Chunk_6 1 0.016853 0.014924 1613.2070 1601.4856 23.4429 6.4339e-07 4.6185e-07 -99 2.5988 4.6776e-03 -99 4 4.00
7 Chunk_7 1 0.014170 0.015146 1611.7855 1601.4856 20.5999 2.8299e-06 1.9182e-06 -99 2.4105 7.9663e-03 -99 2 4.00
8 Chunk_8 1 0.021597 0.019749 1612.6304 1601.4856 22.2896 1.1723e-06 8.2206e-07 -99 2.2307 1.2850e-02 -99 3 3.00
9 Chunk_9 1 0.017626 0.018966 1612.3344 1601.4856 21.6977 1.5959e-06 1.1057e-06 -99 2.5128 5.9890e-03 -99 4 5.00
```

The most important column is number 10, the *p*-values for each gene/chunk (based on a Null Distribution estimated through permutation)

3 - Gene-Based Analysis

The main results file is `chunks/regressALL`

```
> res4=as.matrix(read.table("chunks/regressALL",head=T))
> det4=as.matrix(read.table("chunks/gene_details.txt",skip=3))
> pva4=as.numeric(res4[,10]);chr4=as.numeric(det4[,8])
> plot(-log10(pva4),col=chr4,xlab="SNP",ylab="-log10(P)")
```



3 - Gene-Based Analysis

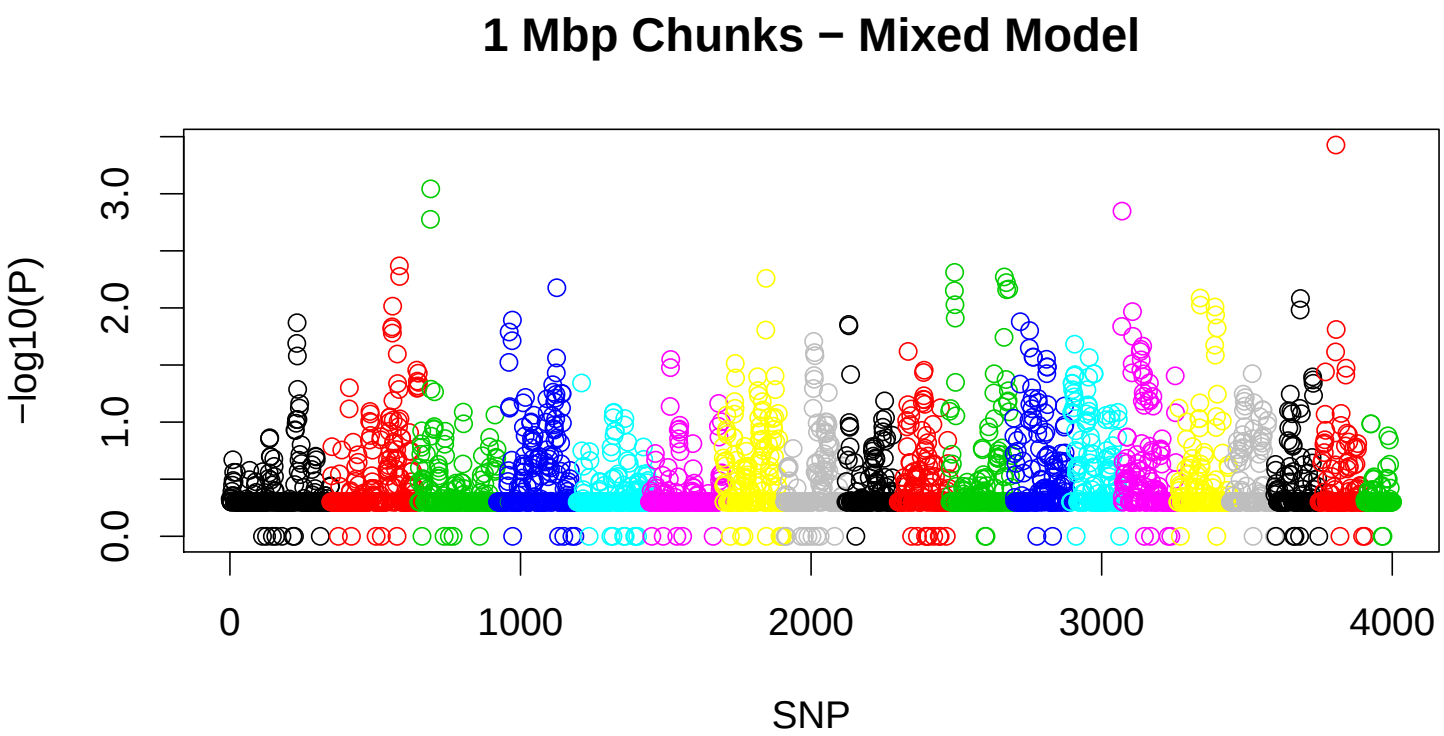
Now repeat, including a grm (same as adding grm to linear model)

```
../ldak.out --cut-genes chunksb --chunks-bp 1000000 \
--bfile mice --ignore-weights YES

../ldak.out --calc-genes-reml chunksb --pheno mice.pheno \
--bfile mice --partition 1 --ignore-weights YES  --mpheno 1 \
--grm micekins

../ldak.out --join-genes-reml chunksb
```

3 - Gene-Based Analysis

The main *p*-values are Column 10 `chunksb/regressALL`

```
> res5=as.matrix(read.table("chunksb/regressALL",head=T))
> det5=as.matrix(read.table("chunks/gene_details.txt",skip=3))
> pva5=as.numeric(res5[,10]);chr5=as.numeric(det5[,8])
> plot(-log10(pva5),col=chr4,xlab="SNP",ylab="-log10(P)")
```



3 - Permutation Analysis

Often useful to repeat the regression using a permuted phenotype. Can do a single permutation by adding `--permute YES`, or multiple permutations by adding (say) `--num-perms 100` and `--fixed-perms YES`

```
../ldak.out --cut-genes chunksc --chunks-bp 1000000 \
--bfile mice --ignore-weights YES
```

```
../ldak.out --calc-genes-reml chunksc --pheno mice.pheno \
--bfile mice --partition 1 --ignore-weights YES  --mpheno 1 \
--grm micekins --num-perms 10 --fixed-perms YES
```

```
../ldak.out --join-genes-reml chunksc
```

4 - MultiBLUP Analysis

Essentially, an extension of gene-based analysis

At the joining step (`--join-genes-reml`) use `--sig1` and `--sig2` to specify significance thresholds for chunks

```
../ldak.out --cut-genes chunksb --chunks-bp 1000000 \
--bfile mice --ignore-weights YES

../ldak.out --calc-genes-reml chunksb --pheno mice.pheno \
--bfile mice --partition 1 --ignore-weights YES  --mpheno 1 \
--grm micekins

../ldak.out --join-genes-reml chunksb --sig1 0.001 \
--sig2 0.01 --bfile mice

ls chunksb/region*
chunksb/region0  chunksb/region2  chunksb/region_details.txt
chunksb/region1  chunksb/region3  chunksb/region_number.txt
```

4 - MultiBLUP Analysis

At the joining step (`--join-genes-reml`) use `--sig1` and `--sig2` to specify significance thresholds for chunks

Then need to create a kinship matrix for all SNPs except those in regions

```
../ldak.out --sub-grm chunksb/region0 --grm micekins \
--bfile mice --region-number 3 --region-prefix chunksb/region
```

```
../ldak.out --reml mblup --pheno mice.pheno --mpheno 1 \
--grm micekins --region-number 3 --bfile mice \
--region-prefix chunksb/region --ignore-weights YES
```

```
../ldak.out --calc-blups mblup --grm micekins \
--remlfile mblup.reml --bfile mice
```

4 - MultiBLUP Analysis

Now instead of standard BLUP (one kinship matrix), run (here) four way BLUP

```
$ head mblup.blup
Predictor A1 A2 Centre Effect
rs3683945 A G 0.886997 0.0001043416
rs3707673 G A 0.887629 0.0000905057
rs6269442 A G 0.719008 0.0001413164
rs6336442 A G 0.887971 0.0000826425
rs13475700 A C 0.262397 -0.0001410037
rs3658242 A T 0.888087 0.0000905841
rs13475701 C G 0.282541 -0.0000866217
rs6198069 A G 0.605982 0.0001568601
rs3659303 A G 0.889004 0.0001118669
```