# GWAS 7: Linkage disequilibrium (LD) and fine-mapping

Matti Pirinen, University of Helsinki

Latest update: 12-Apr-2023



The slide set referred to in this document is "GWAS 7".

### 7.1 Haplotypes

Terms

- **ploidy**, the number of copies of the genome in the cell nucleus.
- **haploid**, one copy of the genome; gametes (sperm and egg cells) are haploid cell types.
- **diploid**, two copies of the genome; other human cell types with nucleus than gametes are diploid.
- **haplotype**, **haploid genotype**, in diploid cell/individual, a genome inherited from one parent.
- **diploid genotype**, a complete set of two genomes of a diploid cell/individual without haplotype information.
- **diplotype**, diploid genotype with haplotype information; a pair of haplotypes for a diploid genotype.

We know that chromosomes are inherited from the parents as recombined segments of grandparents' chromosomes (slide 2). From a genotyping array we observed (diploid) genotypes at each locus, and we can't always be certain which are the two underlying haplotypes (slide 3).

Consider two SNPs 1 and 2 on the same chromosome, with alleles A and C at SNP1 and G and T at SNP2. If we observe an individual whose genotype is SNP1 = A/A, SNP2 = G/T, then her haplotypes are definitely {A-G, A-T}. That is, the two chromosomes that she has inherited from her two parents read A-G and A-T. But if we instead observe genotype SNP1 = A/C, SNP2 = G/T, then the haplotypes are not determined without external information: they can be either {A-G, C-T} or {A-T, C-G}. The process of figuring out the haplotypes at multiple loci from the (unordered) genotypes is called haplotype **phasing**. Historically, the phase was determined from pedigree data where parents' genotypes inform about possible offspring haplotypes. Also experimental methods for phasing exist but are costly. More often, nowadays, phasing is done in large population samples by using computational methods and the result is probabilistic. A review on phasing by Browning & Browning (2011).

How can we phase genotypes into haplotypes? When we consider genotypes at $p$ SNPs where the individual is heterozygous, then he/she has $2^p$ possible haplotypes and $2^{p-1}$ possible phasings (as one haplotype already determines the other haplotype given the genotypes). These numbers become huge already for tens of SNPs. However, empirical data from the early 2000s started to show that when many individuals were genotyped at SNPs residing near each other in the genome, the observed genotypes suggested that only a few ($\ll 2^p$) different haplotypes were present in the population in any one region of the genome.

**Example 7.1.** (Slide 4) Go to LDlink that gives access to the haplotype data from the 1000 Genomes project. Choose LDhap tab and input the following list of 9 SNPs that come from a 2500 bps long region on chr 1:

rs115037027 rs12409788 rs1576517 rs151240271 rs12752436 rs76864380 rs6586443 rs35213023 rs34910942

- Choose first CHS (Southern Han Chinese) population and press `Calculate`. It shows that only two of these SNPs have MAF>5% and four are monomorphic in the Table shown, i.e., show only one allele in these data. There are only 4 haplotypes present in the population out of theoretically possible $2^5 = 32$.
- Change population to FIN. There are 5 haplotypes present and one SNP is monomorphic. Compare SNPs 1 and 6. They have the same pattern of variation across the haplotypes: Whenever one has **T** also the other has **T**. They are completely associated
with each other and, consequently, in a GWAS they would give exactly the same results. Also triple 5, 8 and 9 is similarly fully correlated.
- Change population to LWK (Kenya). Now we see 8 haplotypes, no monomorphic SNPs and only the SNPs 5 and 8 are completely linked with each other.

This example shows how some variants in a narrow region of 2500 bps are highly correlated with their neighbors when it comes to the haplotype patterns that exist in any one population. Typically, African populations show more diversity than Asian and European populations due to more recent genetic bottlenecks that have reduced variation in the history of the latter two. But also in African populations, the haplotype patterns are very much restricted compared to what they would be if the alleles at nearby loci would be distributed nearly independently at the population level.

The human genome has a **haplotype block structure** (slide 5), which suggests that the recombination process shuffles the chromosomes mainly at only a relatively small number of sites on the genome, so called *recombination hotspots*. Between these hotspots, there are larger regions of genome with low recombination rates (slide 6). Consequently,

- Phasing, especially in the low recombination areas, can be done accurately by collecting a large number of genotyped individuals and using a probability model to estimate which are the most likely haplotypes in this population. Since only a few haplotypes exist in the population, we can get a fairly reliable phasing for individuals by sharing haplotype information among them, and we simultaneously also get accurate estimates for the population frequencies of the haplotypes. Existing high quality reference haplotype data from the target population also improves phasing. Phasing software: Beagle5, Eagle2, SHAPEIT5.
- SNPs in the same haplotype block tag each other well because each allele of a particular SNP sits on only a limited number of haplotypes in that block. In particular, by typing the set of the most informative tag-SNPs, we can already cover also a lot of neighboring genetic variation that we don't directly genotype. This was a property that boosted the first GWAS: in order to see statistical associations between genomic regions and phenotypes, it is enough that we manage to genotype tag-SNPs of causal variants.
- The exact causal variants are difficult to pinpoint based on the statistical information alone since so many variants are so highly correlated with each other. This is the **fine-mapping problem**.

### 7.2 Linkage disequilibrium (LD)

We know from empirical data that the human genome has a haplotype block structure. This correlation structure arises because when a new variant (assumed here to be an SNV) is introduced into the population as a mutation, it emerges on one of the existing haplotypes of the population. Because haplotypes are inherited as large segments of grandparental haplotypes, with often just 1 or 2 recombination events per chromosome, the new mutation will be, for many subsequent generations, passed on to the next generation together with the same alleles that were close by to it on its original haplotype background. However, as more generations pass, recombinations cut down the original haplotype background to smaller and smaller shared segments between the descendants of the original haplotype, and hence the correlation at the population level between the variant and its neighbors tends to decrease over time (slides 8-9).

Consider any two SNPs at distinct sites of the genome. Suppose that we had perfect haplotype information. Then we could observe how often their alleles 1 are on the same haplotype in the population and compare that to the expected proportion if the two variants were independently distributed. If allele 1 frequencies in population are $f_k$ and $f_l$ at SNPs $k$ and $l$, and if the frequency of the haplotype 1-1 is $f_{kl}$, then we define

- disequilibrium coefficient $D_{kl} = f_{kl} - f_k \cdot f_l$,

which, theoretically, is 0 if the two loci are independent, also said to be in **linkage equilibrium**, LE. If the loci are not independent ($D_{kl} \neq 0$), then we say that the loci are in **linkage disequilibrium** (LD). Naturally, we never know the frequencies exactly in population and we use sample estimates to compute $D_{kl}$. Hence, we will always observe some non-zero value for $D_{kl}$ in our sample and the interest is in the magnitude of $D_{kl}$.

We can measure the amount of LD also by using the Pearson's correlation coefficient $r_{kl} = \text{cor}(a_k, a_l)$, where $a_k$ and $a_l$ are the indicators of the alleles at SNPs $k$ and $l$ on the same haplotype. It can be shown that $r$ and $D$ are related through

- $r_{kl} = \frac{D_{kl}}{\sqrt{f_k(1-f_k)f_l(1-f_l)}}$.

Note that, theoretically, both $r$ and $D$ are 0 if and only if the loci are in linkage equilibrium. In what follows, we are interested in $r$, because it determines the statistical consequences of LD on GWAS analyses.

A statistical way to think about LD is that if SNPs $k$ and $l$ are in LD, then by observing for a particular haplotype its allele at SNP $k$, we also gain more information about its allele at SNP $l$ than what we had solely based on the population allele frequencies at SNP $l$.

**7.2.1 Generating two-locus haplotype data** Let's follow Vukcevic et al. (2011) who derived formulas for the 4 haplotype frequencies in the population, given the MAFs at the two SNPs (1 and 2), and LD as measured by $r = r_{12}$ between the SNPs. (More comprehensive treatment of the subject is available in Damjan Vukcevic's D.Phil thesis, Oxford, 2009.)

Suppose that SNP1 has alleles **a** (minor) and **A** (major) and SNP2 has alleles **b** (minor) and **B** (major) and that $f_a \leq f_b$. First, in order that the allele frequencies and the given level of LD, as measured by $r$, correspond to a haplotype distribution, the following inequalities must hold:

$$-\sqrt{\left(\frac{f_a}{1-f_a}\right)\left(\frac{f_b}{1-f_b}\right)} \leq r \leq \sqrt{\left(\frac{f_a}{1-f_a}\right) \Big/ \left(\frac{f_b}{1-f_b}\right)}.$$

For example, correlation between two SNPs can be really high ($r \approx 1$) only if both have very similar MAFs, and in order for correlation to be highly negative ($r \approx -1$), both MAFs must be close to 0.5. This is because if we were to assign the minor allele at both SNPs to haplotypes, then the only way that we can make correlation $\approx 1$ is that **a** and **b** are almost always on the same haplotype and **A** and **B** are almost always on the same haplotype. This is only possible if $f_a \approx f_b$ because otherwise there are either some extra **a**s that don't find any **b** to pair with and therefore must pair with **B**, or there are some extra **b**s that don't find an **a** to pair with and therefore must pair with **A**, both of which will lead to a reduction of correlation from $\approx 1$. To get a high negative correlation for the minor alleles, each minor allele must be paired up with the major allele at the other locus, and this is possible only if MAF is close to 0.5 at both loci. Note that if we change the allele coding at one locus from allele 1 being the minor allele to allele 1 being the major allele, that simply changes the sign, but not the magnitude, of the corresponding value of $r$.

The haplotype frequencies, and samples from them, can then be computed using the R-function `geno.2loci()` given below.

```r
geno.2loci <- function(n, r, mafs, return.geno = TRUE){
  #INPUT:
  # n, individuals
  # r, correlation coefficient between the alleles on the same haplotype of the two loci
  # mafs, MAFs of the two loci
  #OUTPUT:
  # if return.geno == TRUE: n x 2 matrix of GENOTYPES of n individuals at 2 loci
  # if return.geno == FALSE: (2n) x 2 matrix of HAPLOTYPES of n individuals (2n haplotypes) at 2 loci
  stopifnot( r >= (-1) & r <= 1 )
  stopifnot( length(mafs) == 2 )
  stopifnot( all(mafs > 0) & all(mafs <= 0.5) )
  stopifnot( n > 0)

  #Label SNPs and alleles so that a and b are minor alleles and freq a <= freq b.
  #At the end, possibly switch the order of SNPs back to the one given by the user.
  f.a = min(mafs) # maf at SNP1
  f.b = max(mafs)  # maf at SNP2

  #With these parameters, the possible LD coefficient r has values in the interval:
  r.min = max( -1, -sqrt(f.a/(1-f.a)*f.b/(1-f.b)) )
  r.max = min( 1, sqrt(f.a/(1-f.a)/f.b*(1-f.b)) )
  #c(r.min,r.max)
  #Check that r is from this interval
  if(r < r.min | r > r.max) stop(paste0("with these mafs r should be in (",r.min,",",r.max,")"))

  # Alleles SNP1: A (major) and a (minor); SNP2: B (major) and b (minor).
  # Compute conditional probabilities for allele 'a' given allele at locus 2:
  q0 = f.a - r*sqrt(f.a*(1-f.a)*f.b/(1-f.b))          #P(a|B)
  q1 = f.a + (1-f.b)*r*sqrt(f.a*(1-f.a)/f.b/(1-f.b)) #P(a|b)

  #Compute the four haplotype frequencies:
  f.ab = f.b*q1
  f.aB = (1-f.b)*q0
  f.Ab = f.b*(1-q1)
  f.AB = (1-f.b)*(1-q0)
  f = c(f.ab,f.aB,f.Ab,f.AB)
  f #These are the haplotype frequencies in the population.
  haps = matrix(c(1,1,1,0,0,1,0,0), nrow = 4, ncol = 2, byrow = T) #4 haplotypes in the population.

  #Generate data for n individuals where each individual is measured at these two SNPs:
  hap.ind = sample(1:4, size = 2*n, replace = T, prob = f) #There are 2*n haplotypes, 2 for each indivi

  if(mafs[1] > mafs[2]) haps = haps[,2:1] #Whether to change the order of loci?
  #Either make genotype matrix by summing the two haplotypes for each individual...
  if(return.geno) X = haps[hap.ind[1:n],] + haps[hap.ind[(n+1):(2*n)],]
  if(!return.geno) X = haps[hap.ind,]  #...or return haplotypes as such.
  return(X)
}
```

Let's check that this works and gives a correlation $\approx r$.

```r
params = matrix(c(0.3,0.3,0.5, #each row has: r, MAF at SNP1, MAF at SNP2
                  0.8, 0.4, 0.3,
```

```
                  -0.5, 0.34, 0.4), byrow = T, ncol = 3)
n = 10000
for(ii in 1:nrow(params)){
  X = geno.2loci(n, r = params[ii,1], mafs = params[ii,2:3], return.geno = FALSE ) #haplotype level dat
  print(c(cor(X[,1],X[,2]), colSums(X)/nrow(X)))
} #and compare to params above
```

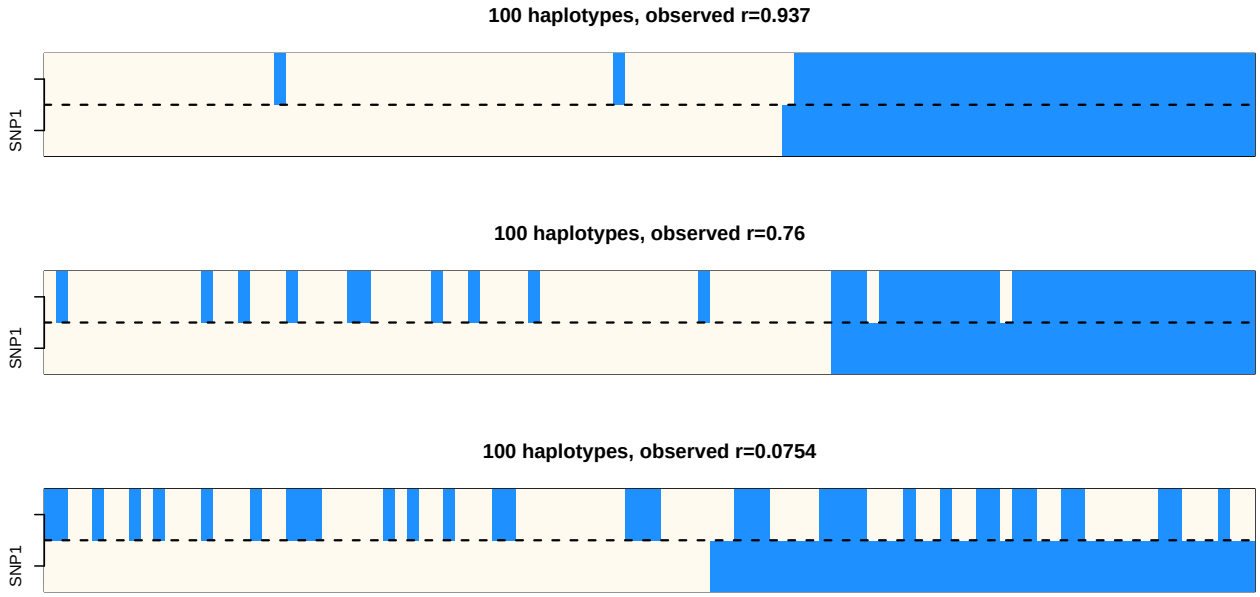
```
## [1] 0.2964545 0.2983500 0.4993500
## [1] 0.8038991 0.3997500 0.3024000
## [1] -0.5016151  0.3387000  0.4026500
```

It works, so let's use it to demonstrate how haplotypes with different LD look like. We start from high LD (r=0.9), then intermediate LD (r=0.66) and finally no LD (r=0). MAFs at the two SNPs must be similar for the high LD case. (Here we use 0.4.)

```
n = 50
rs = c(0.9, 0.666, 0.0) #correlation values
mafs = matrix(c(0.4,0.4, 0.4,0.5, 0.4,0.4), byrow = T, ncol = 2)
cbind(r = rs, mafs) #check that we put correct values in matrix
```

```
##          r
## [1,] 0.900 0.4 0.4
## [2,] 0.666 0.4 0.5
## [3,] 0.000 0.4 0.4
```

```
par(mfrow = c(3,1))
par(mar = c(2,4,4,1))
for(ii in 1:nrow(mafs)){
  X = geno.2loci(n, r = rs[ii], mafs = mafs[ii,],  return.geno = F) #return haplotypes
  X = X[order(X[,1]),] #order so that haps with major allele at SNP1 come first
  image(X, yaxt = "n", xaxt = "n", xlab = "", ylab = "",
        col = c("floralwhite","dodgerblue"), cex.main = 1.3,
        main = paste0(2*n," haplotypes, observed r=",signif(cor(X[,1],X[,2]), 3) ))
  abline(h = 0.5, lwd = 1.5, lty = 2, col = "black")
  axis(2, at = c(0,1), labels = c("SNP1","SNP2"), cex.lab = 1.3 )
}
```




In the first case, we see that allele **A** (white) at SNP1 almost always indicates allele **B** (white) at SNP2, and ALSO allele **a** (blue) indicates allele **b** (blue) at the other SNP. Thus, knowing the allele at either of the SNPs predicts the allele at the other SNP with very high accuracy. These two SNPs are in **high LD**: population parameter was $r = 0.9$, the observed value from the sample is shown in the title.

In the second case, MAF at SNP1 is lower than at SNP2 and even though blue at SNP1 predicts well that SNP2 also has blue allele, also many white alleles at SNP1 go together with a blue allele at SNP2. Hence LD, as measured by $r$, is less than in the first case. This case demonstrates that SNPs with very different MAFs cannot have very high correlation because there is no way to make all blue match blue AND all white match white when there are different numbers of blue and white at the two SNPs. These SNPs are clearly in LD with each other, although not as strongly as the SNPs in the first case.

In the third case, the SNPs are independent in the population ($r = 0$ in simulation), and also in the observed data there is little correlation between the loci. Thus, by knowing the allele at SNP1 does not help predicting what is the allele at SNP2 on the same haplotype. The best predicton we can do statistically is to guess the allele at SNP2 based on the population allele frequencies, and ignore which allele was observed at SNP1, since it does not give additional information over the allele frequencies in population. These SNPs are **not in LD** with each other.

Let's also check what happens to the estimate of $r$ if we estimate it at the genotype level, not at the haplotype level, as we have done so far. Thus, we'll generate haplotypes as previously, but before computing $r$, we collapse the haplotypes into genotypes (by `return.geno = TRUE`) and then compute correlation between the genotypes at the two loci.

```
params = matrix(c(0.3,0.3,0.5, #each row has: r, MAF at SNP1, MAF at SNP2
                  0.8, 0.4, 0.35,
                  -0.5, 0.34, 0.4), byrow = T, ncol = 3)
n = 10000
for(ii in 1:nrow(params)){
  X = geno.2loci(n, r = params[ii,1], mafs = params[ii,2:3], return.geno = TRUE )
  print(c(cor(X[,1],X[,2]), colSums(X)/2/nrow(X)))
} #and compare to params above
```

```
## [1] 0.2793206 0.3005500 0.4936000
## [1] 0.7971017 0.4030500 0.3517000
## [1] -0.4914978  0.3377500  0.4011000
```

This seems to estimate the haplotype level correlations even though the input data are genotypes. This is good to know as the data we observe in practice are at the level of genotypes and phasing it to haplotypes is not always straightforward.

**Example 7.2.** We can also get LD estimates for human populations from LDlink.

- Choose LDhap and consider two SNPs rs7837688, rs4242382 in the CEU population. You'll see the haplotype distribution in 1000G CEU data. There are 3 haplotypes (out of all 4 possible) and one is quite rare (1%). So these are almost in perfect LD where allele at one SNP tells the allele at the other.
- While you could compute $r$ from the info given by LDhap, you can also query it directly from LDpair tab, which gives $r^2 = 0.8791$, i.e., $r = 0.938$. (Slide 10.)
- Look at the same SNPs in LWK population. All 4 haplotypes are present and there is no LD in terms of $r^2$. Often African populations have less LD than non-African since the latter have been through more recent bottlenecks. Such bottlenecks increase LD because when there is a relatively few haplotypes that founded the subsequent generations, then those haplotypes will determine the LD patterns. (Slide 11.)
- Go to LDproxy tab and look for rs7837688 in the Chinese CHB population. You'll see the Figure and list of the variants that are in the highest LD with rs7837688. Note also how the high LD is concentrated within a haplotype block that is between recombination hotspots (region between high peaks of recombination rate cM/Mb). (Slide 12.)
- LDmatrix (slide 13) can be made for a user given set of SNPs.

**Example 7.3.** Let's plot an LD matrix for 74 variants around *APOE* gene on chr 19 using 1000 Genomes Finnish samples (coordinates GRCh37).

```
path = "https://www.mv.helsinki.fi/home/mjxpirin/GWAS_course/material/APOE_1000G_FIN_74SNPS."
haps = read.table(paste0(path,"txt"))
info = read.table(paste0(path,"legend.txt"),header = T, as.is = T)
dim(haps) #rows haplotypes, cols SNPs
```

```
## [1] 186  74
```

```
info[1:3,] #info for first three SNPs
```

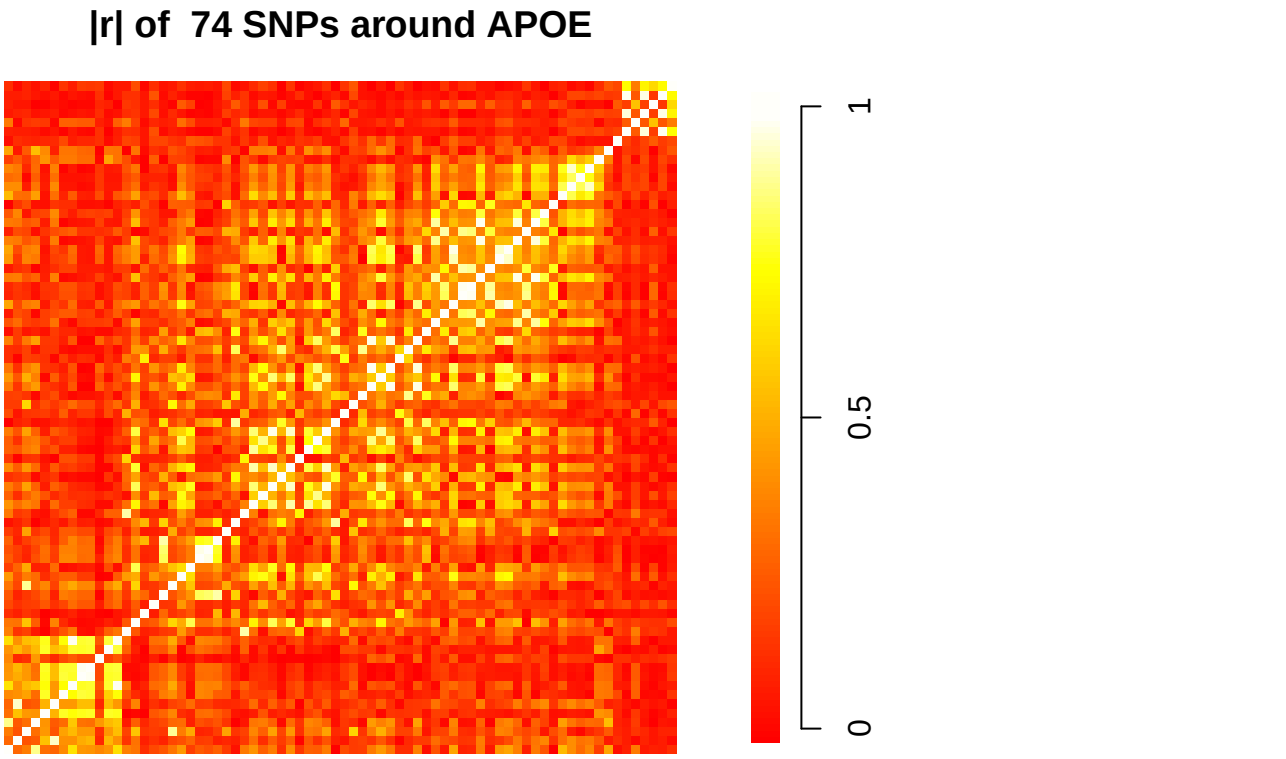
```
##           id position a0 a1    af1
## 1   rs387976 45379060  A  C 0.3065
## 2  rs3852859 45379309  T  C 0.1720
## 3 rs73050293 45379746  A  G 0.2312
```

```
haps[1:3,] #data for first three haplotypes
```

```
##   V1 V2 V3 V4 V5 V6 V7 V8 V9 V10 V11 V12 V13 V14 V15 V16 V17 V18 V19 V20 V21
## 1  0  0  1  0  0  0  0  0  0   0   0   0   0   0   1   0   1   0   1   1   1
## 2  1  0  0  1  1  0  1  1  1   1   0   1   1   0   0   0   1   0   0   1   1
## 3  1  0  0  1  1  0  1  1  1   1   0   1   1   0   0   0   1   0   0   0   0
##   V22 V23 V24 V25 V26 V27 V28 V29 V30 V31 V32 V33 V34 V35 V36 V37 V38 V39 V40
## 1   0   0   0   1   0   0   1   1   1   0   1   0   1   1   1   0   0   1   0
## 2   0   0   0   0   1   0   0   1   0   1   0   0   0   1   0   1   0   0   1
## 3   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0
```

```
##   V41 V42 V43 V44 V45 V46 V47 V48 V49 V50 V51 V52 V53 V54 V55 V56 V57 V58 V59
## 1   1   1   1   0   0   1   0   0   1   1   1   1   0   1   1   1   0   1   0
## 2   1   0   1   0   1   1   1   0   1   1   1   1   0   1   1   1   0   1   0
## 3   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0
##   V60 V61 V62 V63 V64 V65 V66 V67 V68 V69 V70 V71 V72 V73 V74
## 1   0   1   1   0   0   0   1   0   0   0   0   0   0   0   0
## 2   0   1   0   0   0   0   0   1   0   1   0   1   0   1   1
## 3   0   0   0   0   0   0   0   1   0   0   1   0   1   0   1
```

```
R = cor(haps) #LD matrix is the pxp correlation matrix
n.cols = 100
layout(matrix(c(1,2), nrow = 1), widths = c(9,1))
par(mar = c(2,2,3,1)) #plot LD matrix using image
image(abs(R), col = heat.colors(n.cols),
      breaks = seq(0, 1, length = n.cols + 1),
      asp = 1, xaxt = "n", yaxt = "n", bty = "n",
      main = paste("|r| of ", dim(R)[1],"SNPs around APOE"))
par(mar = c(2,1,3,1)) #plot scale for colors
plot.window(xlim = c(0,1), ylim = c(0,n.cols))
points(x = rep(1,n.cols + 1),y=(0:n.cols), col = heat.colors(n.cols + 1), pch = 15, cex = 2)
axis(4, at = c(0, n.cols / 2, n.cols),
     labels = c(0, 0.5, 1) )
```




LD matrices from two Finnish cohorts show highly similar LD structure to each other on slide 14.

### 7.3 Effect of LD on GWAS results

Let's then experiment with a GWAS setting where SNP1 is a causal variant with causal effect size of $\lambda_1 = 0.2$ and MAF $= 0.2$ whereas SNP2 does not have a causal effect ($\lambda_2 = 0$) and has MAF $= 0.4$, on a quantitative

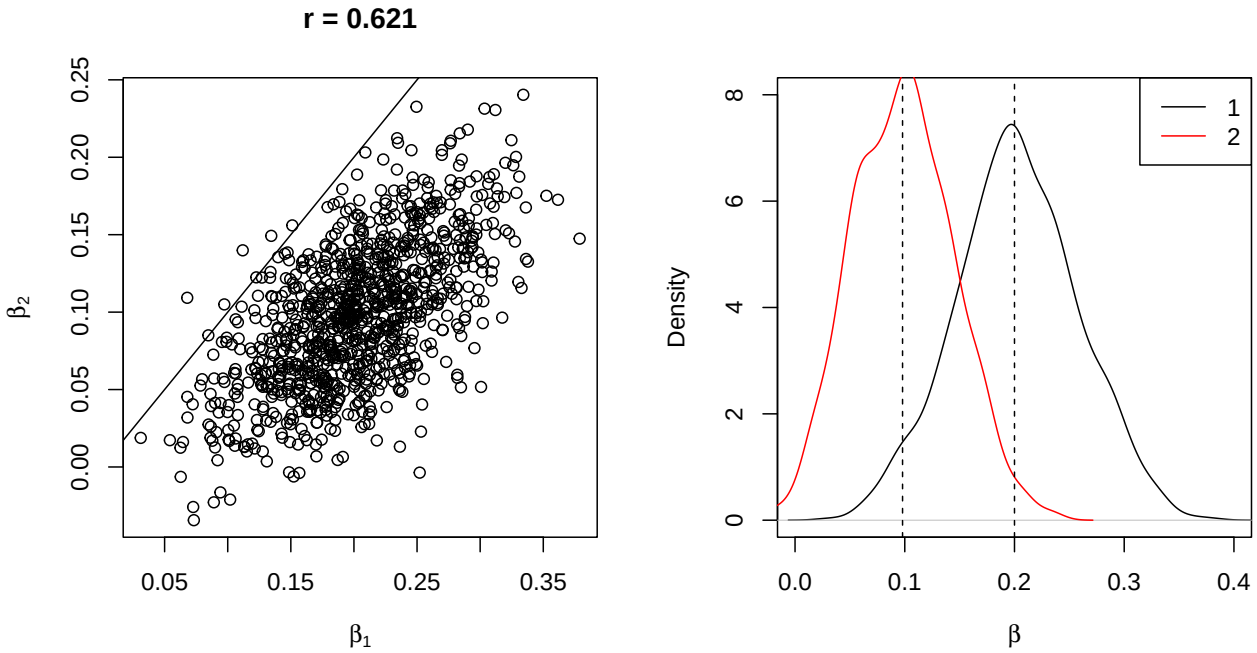
phenotype whose variance is 1 in population. LD between the SNPs is $r_{12} = 0.6$. So far on this course we have used $\beta$ to denote the effect size of allele 1. Reason why we now use $\lambda$ rather than $\beta$ is because we want to separate the **causal effect** ($\lambda$) from the **marginal effect** ($\beta$), as will become clear soon.

Let's make 1000 simulations of such setting, using $n = 1000$ individuals in each simulation, and let's do the typical single-SNP GWAS on each variant and collect the effect estimate $\widehat{\beta}_l$, its SE and P-value for both variants $l = 1, 2$.

```
n.iter = 1000
n = 1000
r = 0.6
mafs = c(0.2, 0.4)
lambda = c(0.2, 0) #causal effects of each SNP
res = matrix(NA, ncol = 6, nrow = n.iter) #6 cols: beta1, SE1, P1; beta2, SE2, P2
colnames(res) = c("beta1","SE1","pval1","beta2","SE2","pval2")
for(iter in 1:n.iter){
  X = geno.2loci(n, r, mafs, return.geno = T) #generate 2-locus genotypes
  y = X %*% lambda + rnorm(n,0, sqrt(1-var(X %*% lambda)) ) #var(y) = 1,
  res[iter, 1:3] = summary(lm(y ~ X[,1]))$coeff[2, c(1,2,4)] #collect beta, SE, P-val of SNP1
  res[iter, 4:6] = summary(lm(y ~ X[,2]))$coeff[2, c(1,2,4)] #collect beta, SE, P-val of SNP2
}
```

Let's plot effect estimates $\widehat{\beta}_1$ and $\widehat{\beta}_2$ against each other and as separate distributions. Let's add vertical lines at the expected mean values: $\lambda_1$ for SNP1 and $r_{12}\,\lambda_1\sqrt{\frac{f_1(1-f_1)}{f_2(1-f_2)}}$ for SNP2 (it will be explained later where this comes from).

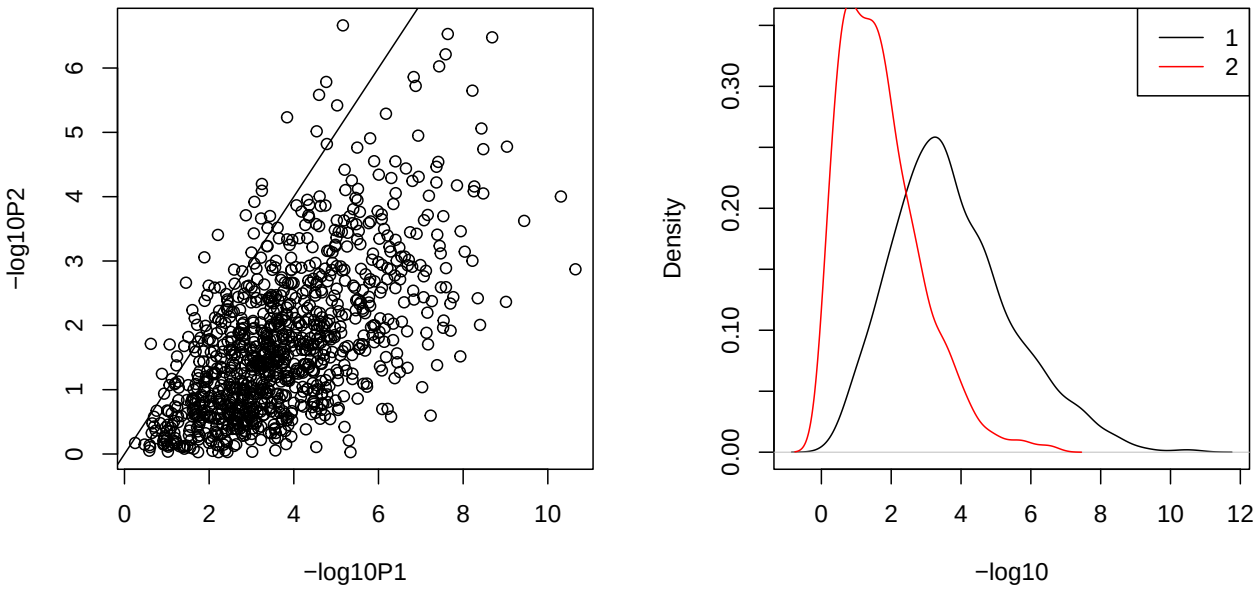
```
par(mfrow = c(1,2))
plot(res[,"beta1"], res[,"beta2"], xlab = expression(beta[1]), ylab = expression(beta[2]),
     main = paste("r =",signif(cor(res[,"beta1"], res[,"beta2"]), 3)))
abline(0,1)
plot(density(res[,"beta1"]), xlab = expression(beta), col = "black", main = "",
     xlim = c(0,0.4), ylim = c(0,8))
lines(density(res[,"beta2"]), col = "red")
abline(v = lambda[1], lty = 2)
abline(v = lambda[1]*r*sqrt(mafs[1]*(1-mafs[1])/mafs[2]/(1-mafs[2])), lty = 2)
legend("topright", col = c("black","red"), lwd = 1, legend = c(1,2) )
```




We see that there is a clear correlation in GWAS effect estimates between SNPs 1 and 2, and that correlation seems to match well with the LD correlation between the SNPs ($r_{12} = 0.6$). The distributions of the effect estimates shows that $\widehat{\beta}_1$ is centered on the true causal effect $\lambda_1$ and $\widehat{\beta}_2$ is centered at value $r_{12}\,\lambda_1\sqrt{\frac{f_1(1-f_1)}{f_2(1-f_2)}}$ (and not at 0, which would be the true *causal* effect of SNP2; derivation on slide 16).

Let's look at the P-values:

```
par(mfrow = c(1,2))
plot(-log10(res[,"pval1"]), -log10(res[,"pval2"]), xlab = "-log10P1", ylab = "-log10P2")
abline(0,1)
plot(density(-log10(res[,"pval1"])), xlab = "-log10", col = "black", main = "", ylim = c(0,0.35))
lines(density(-log10(res[,"pval2"])), col = "red")
legend("topright", col = c("black","red"), lwd = 1, legend = c(1,2) )
```



Also the P-values are correlated and SNP2 has P-values that are clearly much smaller than under the null even though SNP2 does not have a causal effect on the phenotype. We also see that there are a few cases where SNP2 actually has lower P-value (higher -log10 P-value) than SNP1 even though SNP1 is the causal variant that is driving the genotype-phenotype association here. Consequently,

- When SNPs 1 and 2 are LD-friends (in LD with each other), and only SNP1 is causal, then the causal effect of SNP1 will leak to the marginal effect of SNP2 in the sense that the standard one-SNP-at-a-time analysis at SNP2 is estimating the marginal effect

$$\beta_2 = r_{12}\lambda_1\sqrt{\frac{f_1(1-f_1)}{f_2(1-f_2)}},$$

  rather than the true causal effect of SNP2 (which would be 0).

- This means that if we had not genotyped SNP1 but only genotyped SNP2, we would still see the association signal in this region, but its effect size would be reduced by a factor of $r_{12}\sqrt{\frac{f_1(1-f_1)}{f_2(1-f_2)}}$, and its NCP would by reduced by a factor of $r_{12}^2$, compared to those values that we would estimate at the true causal variant.

- Hence, when we see peaks on Manhattan plots, we cannot be sure whether we have the causal variation itself included in our analysis, or whether we are just tagging it by our SNPs. In principle, the possible hidden causal variation does not need to be a single SNV, it can be a structural variant, or a haplotype-level effect, for example.

- The top variant in terms of P-value does not need to be the causal variant because of tagging and statistical sampling effects. In particular, when there are many variants in high LD with the causal(s) then the association region looks messy (slide 15) and we can't know for sure which are the causal variants by looking at the marginal effect sizes and P-values.

**7.3.1 Relation between causal effect $\lambda$ and marginal effect $\beta$** Consider a region with $p$ SNPs with their causal effects in vector $\boldsymbol{\lambda} = (\lambda_1, \ldots, \lambda_p)^T$. This means that the region contributes to the phenotype by $\sum_{l=1}^{p} x_{il}\lambda_l$.

When we do a standard GWAS, that includes one variant at a time in the regression model, we estimate the marginal effects $\beta_l$ rather than the causal effects $\lambda_l$. Above we saw how the non-causal variant ($\lambda_2 = 0$) had non-zero marginal effect size ($\beta_2 \neq 0$), whose value we could predict from the causal effect $\lambda_1$, $r_{12}$ and allele frequencies at the two SNPs. This relationship generalizes to an arbitrary number $p$ of SNPs in a region and the marginal effect at SNP $l$ is

$$\beta_l = \sum_{k=1}^{p} r_{kl}\lambda_k\sqrt{\frac{f_k(1-f_k)}{f_l(1-f_l)}} = \lambda_l + \sum_{k\neq l} r_{kl}\lambda_k\sqrt{\frac{f_k(1-f_k)}{f_l(1-f_l)}},$$

where the first sum is over all variants in the region, and, in the second formulation, we have separated the variant's own causal effect from the sum (as $r = 1$ between the variant and itself) (Slide 17). Thus, if $l$ is not in LD with any other (causal) variant, then the marginal effect $\beta_l$ equals to the causal effect $\lambda_l$. Otherwise, the marginal effect at SNP $l$ combines the causal effect of $l$ with the causal effects from all other variants, weighted by their LD with the SNP $l$. In particular, when a non-causal SNP $l$ is in high LD with one or more causal variants, then we estimate its marginal effect as non-zero even when the SNP has no biologically causal effect on the phenotype. This phenomenon explains why Manhattan plots show piles of nearby SNPs that together reach highly significant P-values: a few of them may be truly causal, but most of them are only in LD with those few causals, and most of the non-zero marginal effects simply reflect the causal effects of their LD-friends.

We want the formulas to be compact and easy to remember, and we are not happy with the above formula with this respect, so we introduce some scaling to help us here.

**Scaled genotypes and scaled effects.** Suppose that MAF at SNP $l$ is $f_l$ and individual $i$ has genotype $x_{il}$. We define **scaled genotype** of individual $i$ as

$$x^*_{il} = \frac{x_{il} - 2f_l}{\sqrt{2f_l(1-f_l)}}.$$

Thus, while $x_{il}$ had three possible values 0, 1 or 2 independent of MAF, also $x^*_{il}$ has three possible values, but these values are chosen so that the average scaled genotype in the population is 0, its standard deviation and variance is 1, and, consequently, the numerical difference between two genotype classes gets larger as MAF gets smaller. This is shown in the Table below, where the columns present the numerical values of the scaled genotypes corresponding to the genotypes 0,1 and 2, for four values of MAF $f$.

| genotype | $f = 0.5$ | $f = 0.10$ | $f = 0.01$ | $f = 0.001$ |
|---|---|---|---|---|
| 0 | -1.41 | -0.47 | -0.14 | -0.045 |
| 1 | 0 | 1.89 | 6.96 | 22.33 |
| 2 | 1.41 | 4.24 | 14.07 | 44.70 |

We further define the scaled effects $\beta^*_l$ by running the (linear or logistic) regression model using the scaled genotypes:

$$Y \sim \mu^* + X^*\beta^*.$$

This model yields exactly the same P-value for $\beta^*_l$ as the regression using original genotypes gives for $\beta_l$, but the effect size that is being estimated is now $\beta^*_l = \beta_l\sqrt{2f_l(1-f_l)}$ and its standard error is $\mathrm{SE}^*_l = \mathrm{SE}_l\sqrt{2f_l(1-f_l)}$ (so z-score $\beta_l/\mathrm{SE}_l = \beta^*_l/\mathrm{SE}^*_l$ doesn't change). **Allelic effect** $\beta_l$ describes the change in phenotype per each copy of allele 1, whereas **scaled effect** $\beta^*_l$ describes change in phenotype per each standard deviation unit of the genotype variable in the population. This linear transformation does not change the statistical model fit ($R^2$ or P-value) at all. In particular, we need not run any new GWAS in order to estimate the scaled effects, but we can simply transform the existing allelic effect estimates $\widehat{\beta_l}$ to the scaled effect estimates $\widehat{\beta^*_l}$ by multiplying by $\sqrt{2f_l(1-f_l)}$. Note also that, for the quantitative traits, the square of the scaled effect ${\beta^*_l}^2 = 2f_l(1-f_l)\beta^2$ is directly the phenotypic variance explained by the SNP.

We do the same with causal effects where **scaled causal effect** $\lambda^*_l = \lambda_l\sqrt{2f_l(1-f_l)}$.

With the scaled effects, the relationship between marginal and causal effects is simply

$$\beta^*_l = \sum_{k=1}^{p} r_{kl}\lambda^*_k = \lambda^*_l + \sum_{k\neq l} r_{kl}\lambda^*_k,$$

where $r_{kl}$ is the correlation between SNPs $k$ and $l$, and that correlation is exactly the same whether we consider original genotypes or the scaled genotypes. Thus we have simplified our earlier formula by replacing the complicated allele-frequency-ratio-square-root by a few *s.

It is clear that when we are interested in the biology behind the phenotype, then we are only interested in the causal variants and their causal effects. How can we estimate these causal effects? So far our GWAS results have only been about marginal effects $\beta$.

**7.3.2 Joint model** Let's write the phenotype as an additive combination of the (causal) effects of all $p$ variants in the genome:

$$y_i = \mu + \boldsymbol{x}_i \cdot \boldsymbol{\lambda} + \varepsilon_i = \mu + \sum_{l=1}^{p} x_{il}\lambda_l + \varepsilon_i,$$

where $\boldsymbol{x}_i$ is the row $i$ of the full genotype matrix $\boldsymbol{X}$ that contains the $p$ genotypes of individual $i$, and $\boldsymbol{\lambda}$ is the vector of the (unknown) causal effects of the $p$ variants. For binary phenotype, $y_i$ is to be interpreted as the log-odds of the disease for individual $i$ (like in logistic regression).

The difference between this **joint model** of all variants and the **marginal model** of a single variant ($y_i = \mu + x_{il}\beta_l + \varepsilon_i$) is that the joint model accounts for the effects of all variants on the phenotype simultaneously whereas the marginal model only considers one variant at a time. Consequently, an estimator of an effect from the marginal model will estimate the marginal effect $\beta_l$ whereas an estimator of the joint model is estimating the causal effects $\boldsymbol{\lambda}$ simultaneously. A standard GWAS applies $p$ separate marginal regression models, one for each variant, and if we collect the marginal effects into a single vector $\boldsymbol{\beta} = (\beta_1, \ldots, \beta_L)^T$, then by the formulas above, we have a linear relationship between the marginal and causal effects:

$$\boldsymbol{\beta^*} = \boldsymbol{R\lambda^*} \quad \text{or equivalently} \quad \boldsymbol{\lambda^*} = \boldsymbol{R}^{-1}\boldsymbol{\beta^*},$$

where $\boldsymbol{R}$ is the $p \times p$ **LD-matrix** whose element $(l, k)$ is the pairwise correlation $r_{lk}$ of variants $l$ and $k$. We give this formula for scaled effects rather than the allelic effects because the same formula for allelic effects involves ratios of allele frequencies for each pair of variants, which would complicate the notation compared to this version where the differences in allele frequencies have been scaled away. But remember that we can always get from the scaled effect of variant $l$ to the allelic effect of $l$ by dividing by $\sqrt{2f_l(1-f_l)}$.

These considerations suggest two ways to estimate the causal effects $\boldsymbol{\lambda}$.

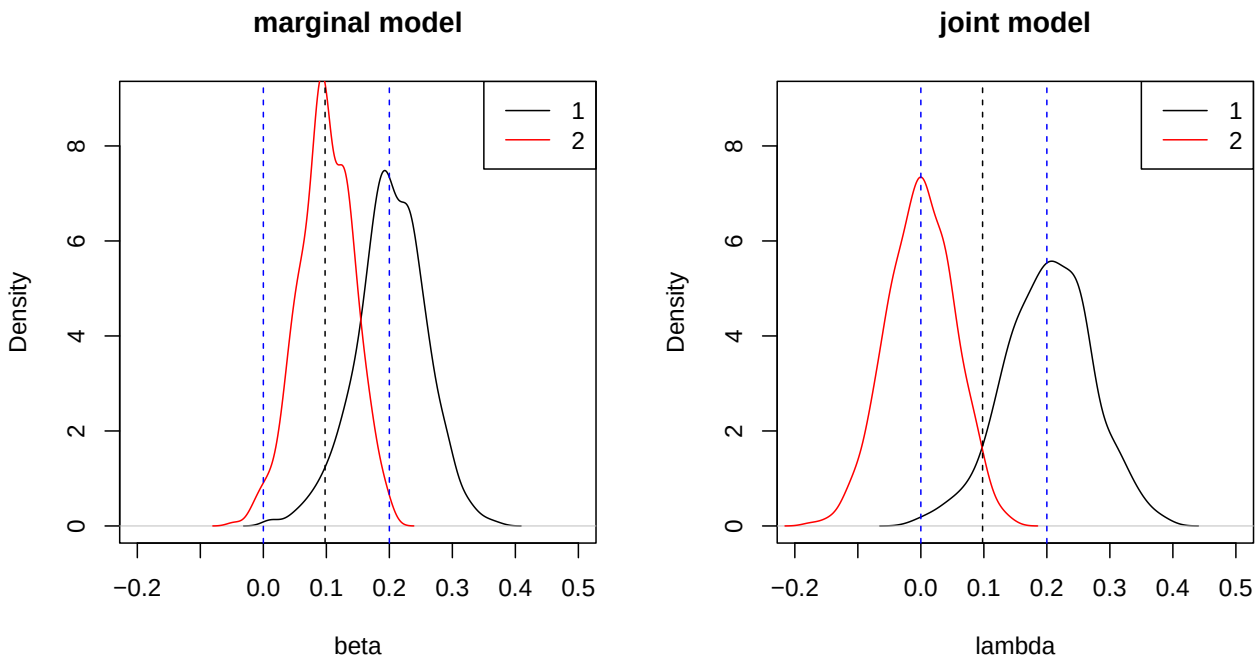
1. Use a multiple regression model where phenotype is regressed on multiple genotypes simultaneously. In R, this is done by simply using a matrix $X$ of all genotypes in the formula like `lm(y~X)` or `glm(y~X, family="binomial")`.
2. Take the marginal GWAS effects and transform them by the inverse of the LD matrix: $\boldsymbol{\lambda^*} = \boldsymbol{R}^{-1}\boldsymbol{\beta^*}$.

Let's try the first one with the settings of our previous example, where we had MAFs 0.2 and 0.4, causal effects $\boldsymbol{\lambda} = (0.2, 0)$ and $r_{12} = 0.6$ in the population. Let's fit both the marginal models and the joint model and plot the estimates of $\beta$s and $\lambda$s.

```
n.iter = 500
n = 1000
r = 0.6
mafs = c(0.2, 0.4)
lambda = c(0.2, 0) #causal effects of each SNP
res = matrix(NA, ncol = 12, nrow = n.iter)
colnames(res) = c("beta1","SE.b1","pval.b1","beta2","SE.b2","pval.b2",
                  "lambda1","SE.l1","pval.l1", "lambda2","SE.l2","pval.l2")
for(iter in 1:n.iter){
  X = geno.2loci(n, r, mafs, return.geno = T) #generate 2-locus genotypes
  y = X %*% lambda + rnorm(n, 0, sqrt(1 - var(X %*% lambda)) ) #var(y) = 1,
  res[iter, 1:3] = summary(lm(y ~ X[,1]))$coeff[2, c(1,2,4)] #collect beta, SE.b1, P.b1 of SNP1
  res[iter, 4:6] = summary(lm(y ~ X[,2]))$coeff[2, c(1,2,4)] #collect beta, SE.b1, P.b1 of SNP2
  joint.coeff = summary(lm(y ~ X))$coeff
  res[iter, 7:9] = joint.coeff[2, c(1,2,4)] #collect lambda1, SE.l1, P.l1 of SNP1 from joint model
  res[iter, 10:12] = joint.coeff[3, c(1,2,4)] #collect lambda2, SE.l2, P.l2 of SNP2 from joint model
}

par(mfrow = c(1,2))
#1st row (2 panels):
#Plot marginal effects as separate distributions
#Add blue lines to the true causal effects lambda
#Add black line to the expected marginal effects of beta2:
#r*sqrt(maf1*(1-maf1)/maf2/(1-maf2))*lambda1
for(ii in 1:2){
  eff = c("beta","lambda")[ii]
  title.txt = c("marginal model", "joint model")
```

```
  plot(density(res[,paste0(eff,"1")]), xlab = eff, col = "black",
       main = title.txt[ii], ylim = c(0,9), xlim = c(-0.2,0.5))
  lines(density(res[,paste0(eff,"2")]), col="red")
  abline(v = lambda[1], lty = 2, col = "blue")
  abline(v = lambda[2], lty = 2, col = "blue")
  abline(v = lambda[1]*r*sqrt(mafs[1]*(1-mafs[1])/mafs[2]/(1-mafs[2])), lty = 2, col = "black")
  legend("topright", col = c("black","red"), lwd = 1, legend = c(1,2) )
}
```




We see that the estimates from the joint model are centered on the true causal effects (blue lines, right panel), whereas the marginal effect of the non-causal SNP2 is estimating the value $r_{12}\,\lambda_1\sqrt{\frac{f_1(1-f_1)}{f_2(1-f_2)}}$ (black line, left panel).

It also seems that the distributions of estimates are wider for $\lambda$s than for $\beta$s. That is because when the two variables are correlated, the regression model tries to figure out how to divide the total effect between the two correlated variables, which increases uncertainty about how large the effect should be *for any one of them separately*. This uncertainty becomes larger the higher the magnitude of the correlation. At the extreme, for two perfectly correlated variables, there is no way to separate their effects statistically; the standard regression model breaks down if you try to include the same variable twice there.

The price to pay for an unbiased estimate of the causal effect by the joint model is an increased uncertainty of the effect estimate compared to the marginal model. It can be shown that

$$\mathrm{SE}\left(\widehat{\lambda_1}\right) = \frac{\sigma_\varepsilon}{\sqrt{2n\,(1-r_{12}^2)\,f_1(1-f_1)}} \approx \frac{\mathrm{SE}\left(\widehat{\beta_1}\right)}{\sqrt{(1-r_{12}^2)}}.$$

Thus, when SNPs are independent ($r_{12}=0$), the precision of $\widehat{\lambda}$s is the same as for $\widehat{\beta}$s and when correlation approaches $\pm 1$, then the joint model becomes more and more unsure how the joint effect should be split between the two candidates and SE of $\widehat{\lambda}$s grows towards infinity. In practice, it is not useful to include very highly correlated variables in a regression model simultaneously because their effects cannot be separated from each other (except with extreme amounts of data).

It follows that if we look at the P-value from the Wald statistic of SNP1, then we would expect to see lower

P-values for $\beta_1$ than for $\lambda_1$ because while both estimate the correct causal effect 0.2, SE of $\widehat{\beta}_1$ is lower than SE of $\widehat{\lambda}_1$, which leads to lower P-values for $\beta_1$ than for $\lambda_1$ on average.

This is not a problem in standard GWAS, where we first identify the regions where variants show high association based on the marginal models, ($\widehat{\beta}$s and their P-values), and then we turn to a more detailed analysis of those regions using some versions of a joint model.

A joint (linear or logistic) regression model of several SNPs gives unbiased estimates of the causal effects of the SNPs assuming that ALL causal variants of the region are included in the model. This is a nice property, but in the same time these standard regression models are not suitable for large scale variable selection among thousands of candidate variables and they do not handle well highly correlated variables. Therefore, we can only use them for joint models of a handful of variants at a time. Next we will look how this is typically done for a GWAS region by stepwise forward selection. And later we will see how more recent fine-mapping models combine sparse causal models with efficient search algorithms.

### 7.4 Stepwise forward selection

Let's aim to reduce the set of $p$ SNPs in the region into a (much) smaller subset $S$ of SNPs that can already statistically explain the association pattern that we see in the region. Technically, this means that we want to find as small a subset $S$ as possible, for which the joint regression model

$$Y \sim \boldsymbol{X}_S$$

explains the data $Y$ already so well that any extended model

$$Y \sim \boldsymbol{X}_S + X_a,$$

with one (or more) additional SNP(s) $a \notin S$ included, would not lead to a statistically better fit to the observed data. (Here $\boldsymbol{X}_S$ is a matrix whose columns contain the genotypes of the SNPs included in set $S$.)

There could be many versions of such a set $S$, and here we start by trying to find one of them.

We will demonstrate this with two data sets generated using the *APOE* region data on 186 Finnish haplotypes from the 1000 Genomes project, and we call these data sets as APOE.1 and APOE.2.

**Example 7.4: APOE.1 data.** Let's generate 1000 individuals by sampling their haplotype pairs from the reference pool with replacement and adding them up to get individual level genotypes. This reflects what we would observe from genotyping.

```
path = "https://www.mv.helsinki.fi/home/mjxpirin/GWAS_course/material/APOE_1000G_FIN_74SNPS."
haps = as.matrix(read.table(paste0(path,"txt")))
dim(haps) #rows haplotypes, cols SNPs
```

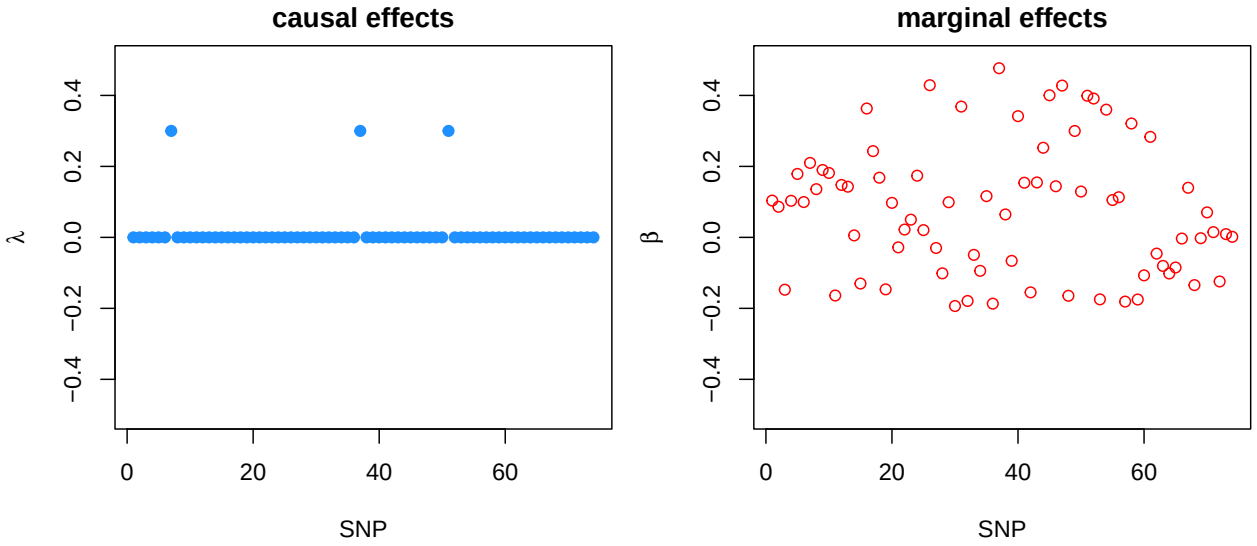
```
## [1] 186  74
```

```
n = 1000
X = haps[sample(1:nrow(haps), size = n, repl = T),] + haps[sample(1:nrow(haps), size = n, repl = T),] #
```

Let's then choose three causal variants: 7, 37 and 51, and give them all a causal effect of $\lambda = 0.3$. Let's plot the causal and marginal effects for the region.

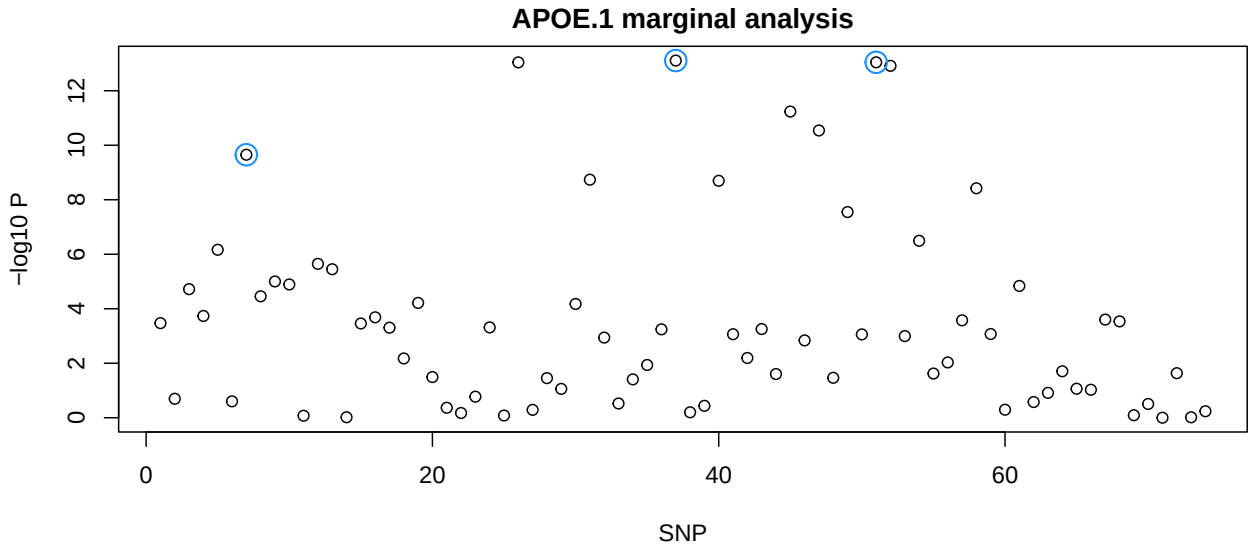
```
c.ind.1 = c(7, 37, 51) # causal SNP indexes for APOE.1 data
p = ncol(X)
lambda = rep(0, p)
lambda[c.ind.1] = 0.3
```

```
f = colSums(X)/2/nrow(X) # allele 1 freqs
f.sc = sqrt(2*f*(1-f)) #scaling factor for genotypes
R = cor(X) # R is computed from genotypes
beta = (R %*% (lambda * f.sc)) / f.sc # from lambdas to betas
par(mfrow = c(1,2))
par(mar = c(4.5, 4.5, 2, 0.5))
plot(lambda, xlab = "SNP", ylab = expression(lambda), col = "dodgerblue",
     pch = 19, ylim = c(-0.5,0.5), main = "causal effects")
plot(beta, xlab = "SNP", ylab = expression(beta), col = "red",
     pch = 1, ylim = c(-0.5,0.5), main = "marginal effects")
```




We see that many marginal effects are larger than any causal effects because some SNPs are in high LD with several causals and therefore their marginal effects are a weighted sum of the causals. In particular, some non-causals have larger marginal effects than some causals, and therefore it could happen that the top-SNP in this region is not a causal variant. Let's generate a trait with variance 1, do a standard marginal GWAS model, and highlight the causal variants in the Manhattan plot of this region.

```
set.seed(18)
y.1 = X %*% lambda + rnorm(n, 0, sqrt(1 - var(X %*% lambda)) ) #trait for APOE.1 data
res.1 = apply(X, 2, function(x){summary(lm(y.1 ~ x))$coeff[2,c(1,2,4)]}) #3 rows: beta,SE,pval
par(mar = c(4.5, 4.5, 2, 1))
plot(-log10(res.1[3,]), xlab = "SNP", ylab = "-log10 P", main="APOE.1 marginal analysis")
points(c.ind.1, -log10(res.1[3,c.ind.1]), cex = 2, lwd = 1.4, col = "dodgerblue") #highlight causals
```

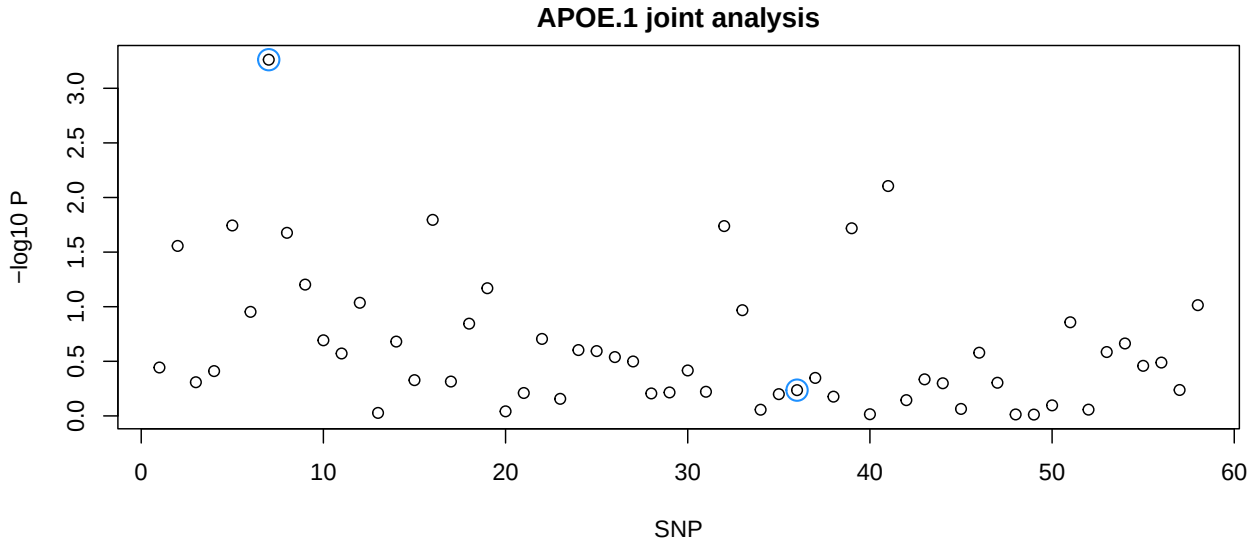



The top variant is one of the causals (37) but some non-causal variants are also nearby with similar P-values. Could we fit a joint model for all the variants in the region and get a clearer picture of the causal ones?

```
joint.coeff = summary(lm(y.1 ~ X))$coeff[-1, c(1,2,4)] #joint model isn't able to return results for al
nrow(joint.coeff) # how many coefficients did we get out of 74?
```

```
## [1] 58
```

```
par(mar = c(4.5, 4.5, 2, 1))
plot(-log10(joint.coeff[,3]), xlab = "SNP", ylab = "-log10 P", main = "APOE.1 joint analysis")
ring.i = which(row.names(joint.coeff) %in% paste0("XV",c.ind.1)) # indexes of the causals present
points(ring.i, -log10(joint.coeff[ring.i,3]), cex = 2, lwd = 1.4, col = "dodgerblue")
```




Unfortunately, the standard linear regression is unable to compute statistics for 16 out of 74 variants because of high correlations – we miss also one of the causals – and the joint model gives nearly useless results for the remaining SNPs as the effects are split between all correlated SNPs simultaneously, whereas in reality there are only three causals here. Technically, the joint model would work, if we had a large enough sample size, but for many SNPs with high correlations, those sample sizes would need to be unrealistically large.

A more robust way is to use **stepwise forward selection** (also called **iterative conditioning**, or **conditional analysis**) to build iteratively a joint model for a small subset $S$ of SNPs by choosing one SNP at a time to be included in $S$. This approach was made popular by GCTA's Conditional & joint (COJO) analysis of GWAS results. The algorithm works as follows

```
initially S is empty
repeat until all P-values outside S are > threshold
  add SNP with the lowest P-value to S
  update P-values of all SNPs 'l' outside S using joint model Y ~ X.S + X.l
end repeat
```

After the algorithm finishes, $S$ contains a set of SNPs whose P-values in the joint model with their predecessors in $S$ are below the given threshold, and there are no additional SNPs with this property outside $S$.

Idea is that $S$ will only contain one of the SNPs that are tagging similarly the causals because when two such SNPs are simultaneously in the model, then they don't have low P-values anymore, which is saying that the second one is not needed *after the first one is already in the model*. This *conditioning* on the current contents of $S$ gives the procedure the name of iterative conditioning / conditional analysis. The name we use here, *stepwise forward selection*, means that we proceed *forwards* from the empty model, step-by-step, where at each step one SNP can be selected to be included in S. (*Backward selection* would start with all variables in the model and would proceed by eliminating them one-by-one. This would not be a feasible strategy when there are a lot of candidate variables available.)

On the other hand, if there are two causal variants in the region, then the P-value of a SNP tagging primarily the second causal variant will remain low after we condition on the top-SNP that tags primarily the first causal signal. Hence both SNPs may be included in $S$. Sometimes the P-value of the second SNP may even get much lower after the first SNP is included in the model, if the two SNPs were masking each other in their marginal effects. Slide 18 has an example of such a case from a Parkinson's disease locus 4q22/*SNCA*.

Thus, when stepwise selection includes several variants in $S$, it suggests that either there are multiple causal variants in the region, or that at least we need more than one variant to tag well a single causal variant in the region, e.g., because the actual causal variant is not genotyped.

Let's write a function that does the stepwise selection and produces a plot of P-values after each step

- showing the newly chosen SNP in red,
- showing the preceding SNPs in $S$ in gray,
- highlighting the true causals with blue.

```
stepwise.fwd <- function(y, X, p.thresh = 1e-4, plot.path = T, ring.i = NULL){
  #Does stepwise forward selection where at most one variant is included in S at each iteration
  ## INPUT
  # y, quantitative trait for n individuals
  # X, genotypes for n individuals x p SNPs
  # p.thresh, P-value threshold that is used for deciding whether to include variant in S
  # plot.path, TRUE / FALSE, whether produces a plot of every step of the algorithm
  # ring.i, set of indexes of SNPs that should always be highlighted in plots by a ring
  ## OUTPUT
  # chosen.i, indexes of chosen variants in S,
  # chosen.p, the P-values at the iteration when each chosen variant was chosen,
  # next.p, the smallest P-value left outside S when finished.

  p = ncol(X) #number of SNPs
```

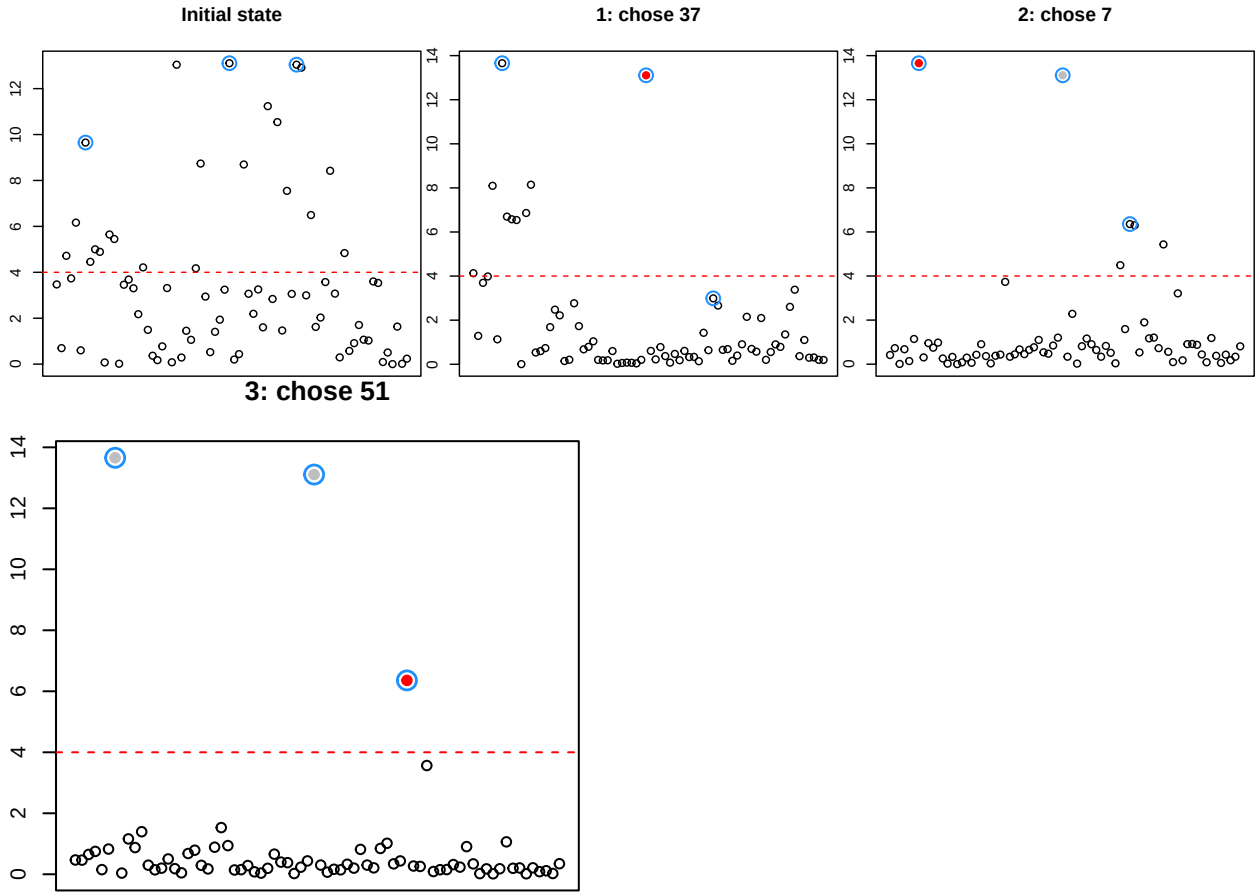
```
  col.new = "red"
  col.old = "gray"
  col.ring = "dodgerblue"
  pval = apply(X, 2, function(x){summary(lm(y ~ x))$coeff[2,4]}) #start from marginal P-values
  cols = rep("black", p)
  pchs = rep(1, p)
  chosen.i = c() #collect here the chosen SNPs
  chosen.p = c() #collect here the P-values from the iteration where the choice was made
  if(plot.path) {
    par(mfrow = c(1, 3))
    par(mar = c(1, 2, 4, 0.1))
    txt = "Initial state"
    plot(-log10(pval), xaxt = "n", xlab = "", ylab = "-log10P",
         main = txt, col = cols)
    if(!is.null(ring.i)) points(ring.i, -log10(pval[ring.i]), cex = 2, lwd = 1.4, col = col.ring) # hig
    abline(h = -log10(p.thresh), lty = 2, col = "red")
  }
  ii = 0 #iteration index
  while( min(pval) < p.thresh ){ #continue as long as something is below the threshold
    ii = ii + 1
    chosen.i[ii] = which.min(pval)[1] # add 1 SNP with min P-val to the chosen ones...
    chosen.p[ii] = pval[chosen.i[ii]] #... and store its P-value at this iteration.
    #test other SNPs except the already chosen ones -- and include the chosen ones as covariates
    tmp = apply(X[,-chosen.i], 2, function(x){summary(lm(y ~ x + X[,chosen.i]))$coeff[2,4]})
    pval[-chosen.i] = tmp # we have P-values for other SNPs except already chosen ones in 'tmp'
    if(plot.path) {
      pval[chosen.i] = chosen.p # add causals with P-values temporarily for plotting
      cols[chosen.i] = col.old # earlier chosen remain gray --
      cols[chosen.i[ii]] = col.new # -- the one newly chosen will be red
      pchs[chosen.i[ii]] = 19 #solid for chosen ones
      txt = paste0(ii,": chose ",chosen.i[ii])
      plot(-log10(pval), xaxt = "n", xlab = "", ylab = "-log10P",
           main = txt, col = cols, pch = pchs)
      if(!is.null(ring.i)) points(ring.i,-log10(pval[ring.i]), cex = 2, lwd = 1.4, col = col.ring)
      abline(h = -log10(p.thresh), lty = 2, col = "red")
    }
    pval[chosen.i] = 1 #mark chose ones by P-value = 1 for the while-loop condition
  }
  return(list(chosen.i = chosen.i, chosen.p = chosen.p, next.p = min(pval)))
}
```

We use a fairly liberal threshold of P=1e-4 here for two reasons:

1. We want the set $S$ to fully explain the association pattern in the region so that no peaks of P-values remain.
2. When we know that there is a robust GWS association in the region, then the variants in the region have a higher prior probability to be associated than random variants in the genome.

There is no consensus value to use at stepwise conditioning. Sometimes it is run with the GWS-level of 5e-8, and sometimes continued until $P > 0.05$. Luckily, the path of the algorithm doesn't depend on the threshold used, so one can also cut the path later to be more stringent as needed for any particular application.

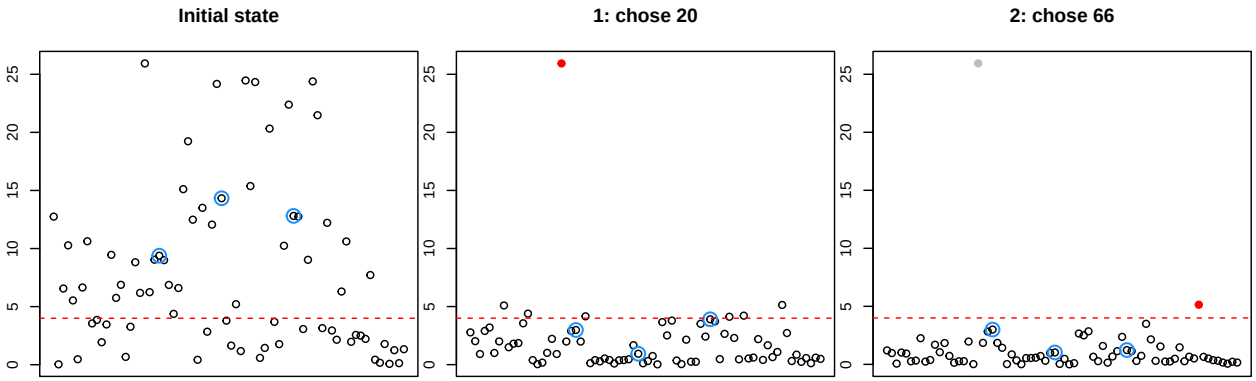
```
step.1 = stepwise.fwd(y.1, X, p.thresh = 1e-4, plot.path = T, ring.i = c.ind.1)
```




Here the stepwise selection worked well in the sense that it managed to pick exactly the three causal variants. Note also how the P-values of true causal variants 7 and 51 get smaller in steps 1 and 2, respectively, compared to the P-values from the previous iteration. This shows how conditioning on the other causal variants can reveal the true causal effects more clearly.

**Example 7.5: APOE.2 data.** Stepwise forward selection doesn't always work quite so nicely:

```
set.seed(20)
c.ind.2 = c(23,36,51)
lambda = rep(0, p)
lambda[c.ind.2] = 0.4
y.2 = X %*% lambda + rnorm(n, 0, sqrt(1-var(X %*% lambda)) )
res.2 = apply(X, 2, function(x){summary(lm(y.2 ~ x))$coeff[2,c(1,2,4)]})
step.2 = stepwise.fwd(y.2, X, p.thresh = 1e-4, plot.path = T, ring.i = c.ind.2)
```




Let's see how the chosen variants 20 and 66 are related to the causal ones 23, 36 and 51:

```
ind = c(step.2$chosen.i, c.ind.2)
R[ind,ind]
```

```
##            V20         V66         V23         V36        V51
## V20  1.0000000 -0.13799708  0.31314083  0.63185564  0.3793202
## V66 -0.1379971  1.00000000 -0.03857787 -0.10127858  0.3980259
## V23  0.3131408 -0.03857787  1.00000000 -0.05261768  0.2210782
## V36  0.6318556 -0.10127858 -0.05261768  1.00000000 -0.1160176
## V51  0.3793202  0.39802586  0.22107824 -0.11601760  1.0000000
```

It seems that none of the causals is very highly tagged by either of the chosen ones, but all causals show considerable correlation with the chosen ones. Let's check whether the chosen set explains the phenotype similarly to the causal variants.

```
lm.chosen = lm( y.2 ~ X[,step.2$chosen.i]) #chosen SNPs 20 and 66
summary(lm.chosen)
```

```
##
## Call:
## lm(formula = y.2 ~ X[, step.2$chosen.i])
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -2.63463 -0.65461 -0.00045  0.64730  2.97895
##
## Coefficients:
##                           Estimate Std. Error t value Pr(>|t|)
## (Intercept)              -0.14676    0.07391  -1.986   0.0473 *
## X[, step.2$chosen.i]V20   0.51863    0.04461  11.626  < 2e-16 ***
## X[, step.2$chosen.i]V66   0.21270    0.04717   4.510 7.27e-06 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.9416 on 997 degrees of freedom
## Multiple R-squared:  0.126,  Adjusted R-squared:  0.1243
## F-statistic: 71.88 on 2 and 997 DF,  p-value: < 2.2e-16
```

```
lm.causal = lm( y.2 ~ X[,c.ind.2]) #causal SNPs 23, 36, 51
summary(lm.causal)
```

```
##
## Call:
## lm(formula = y.2 ~ X[, c.ind.2])
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -2.66228 -0.62076  0.02548  0.62554  2.79913
##
## Coefficients:
##                 Estimate Std. Error t value Pr(>|t|)
## (Intercept)      0.01011    0.05740   0.176     0.86
## X[, c.ind.2]V23  0.32885    0.06139   5.357 1.05e-07 ***
## X[, c.ind.2]V36  0.40252    0.04238   9.498  < 2e-16 ***
## X[, c.ind.2]V51  0.38939    0.05150   7.561 9.05e-14 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.9279 on 996 degrees of freedom
## Multiple R-squared:  0.1521, Adjusted R-squared:  0.1495
## F-statistic: 59.54 on 3 and 996 DF,  p-value: < 2.2e-16
```

The chosen variants explain about 12.4% (R^2-adj.) of the trait variance whereas the three causals together explain about 15.0%. Note that we compare the adjusted-R^2 measure which attempts to take into account the number of parameteres in the models whereas multiple-R^2 tends to always favor model with more predictors. Another way to compare models with possibly different numbers of parameters is the Bayesian Information Criterion (BIC) that penalizes models for additional parameters. In short, a lower value of BIC suggests a better model.

```
cbind(chosen = BIC(lm.chosen), causal = BIC(lm.causal))
```

```
##        chosen   causal
## [1,] 2742.155 2718.804
```

We see that also BIC favors the true causal model over the one chosen by the stepwise selection (23 units in BIC is a clear difference).

What if we put all 5 variants in the same model?

```
lm.joint = lm( y.2 ~ X[,ind]) #all 5 SNPs
summary(lm.joint)
```

```
##
## Call:
## lm(formula = y.2 ~ X[, ind])
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -2.71039 -0.62873  0.02886  0.65192  2.77286
##
```

```
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) -0.08465    0.07510  -1.127  0.25992
## X[, ind]V20  0.13341    0.08361   1.596  0.11088
## X[, ind]V66  0.09648    0.05591   1.726  0.08470 .
## X[, ind]V23  0.29795    0.06715   4.437 1.01e-05 ***
## X[, ind]V36  0.31853    0.06918   4.604 4.68e-06 ***
## X[, ind]V51  0.28384    0.07490   3.790  0.00016 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.927 on 994 degrees of freedom
## Multiple R-squared:  0.1554, Adjusted R-squared:  0.1511
## F-statistic: 36.57 on 5 and 994 DF,  p-value: < 2.2e-16
```

We see that, in this joint model, the two non-causals (20,66) picked by conditional analysis are not needed in the model once we have the causals there. (This model explains 15.1% while model with 3 causals already explained 15.0%.)

We conclude that even though the stepwise selection picked a model that had a reasonable explanatory power, the model comparison shows that the chosen model has a statistically worse fit than the true causal model. Let's next move to recent work on fine-mapping methods to see how we can improve the search and quantification of causal variants.

### 7.5 Fine-mapping

When we **fine-map** a GWAS region, our goal is to identify the set of causal variants $C$. Since high levels of LD among candidate variants often makes it impossible to give a definite answer based on statistical evidence alone, our answers will be probabilistic. In particular, we work with

- probability for variant $l$ to be one of the causal variants in the region, $p_l = \Pr(l \in C \,|\, \text{Data})$. This is also called **posterior inclusion probability (PIP)**.
- probability for a set $S$ of variants to be the **causal set / causal configuration** of variants in the region, $p_S = \Pr(S = C \,|\, \text{Data})$.
- **credible sets** of causal variants which have the property that, with some fixed coverage, such as 95% probability, the credible set contains one of the causal variants or the credible set contains all of the causal variants, where the choice between these two definitions of the credible set depends on the context.
- probability distribution that the region contains $1, 2, \ldots K$ causal variants up to some value of $K$, typically $\leq 20$.

Note the difference between such quantities and the results from stepwise selection, which only resulted in one set of variants, but did not estimate how likely each of those variants is to be a causal variant *given the LD patterns and other candidate variants in the region.* Even if a variant has very low marginal P-value, if it shares the signal with 10 other highly correlated variants, then its PIP will likely be $\leq 0.1$ and its credible set will contain all its 10 LD-friends. Neither of these properties can be observed from the output of stepwise selection.

A review of fine-mapping methods is given by Schaid et al. A common theme is the use of a Bayesian model that allows quantifying the probabilities of causality for each set and each variant, under the assumptions that all causal variants are included in the analysis and that there are no interaction effects between the variants.

Slides 21-30 have some more details of the model behind a widely-used FINEMAP software, written by Christian Benner while he did his PhD thesis at FIMM, Helsinki. Another popular fine-mapping method is SuSiE.

Let's see what happens when FINEMAP is run on our two data sets APOE.1 and APOE.2 where we observed varying success with stepwise selection.

**FINEMAP on APOE.1** FINEMAP gives three output files:

- **.snp** includes summary data (probabilities and causal effect sizes) for each SNP.
- **.config** includes list of most probable sets of causal variants, with their probabilities and variance explained.
- **.cred3** includes credible sets for 3 causal variants corresponding to the top-configuration in .config file.

Let's read in FINEMAP's results for individual SNPs and print some columns.

```
fm.path = "https://www.mv.helsinki.fi/home/mjxpirin/GWAS_course/material/FINEMAP/FINEMAP_APOE_"
fm.1.snp = read.table(paste0(fm.path,"1.snp"), as.is = T, header = T)
fm.1.snp[1,]
```

```
##   index rsid chromosome position allele1 allele2 maf     beta        se       z
## 1     7    7         19 45382034       A       G 0.5 0.289445 0.0451505 6.41067
##       prob log10bf     mean        sd mean_incl   sd_incl
## 1 0.999999 7.92959 0.324947 0.0703492  0.324947 0.0703488
```

```
fm.1.snp[1:10,c(1,11,13:16)]
```

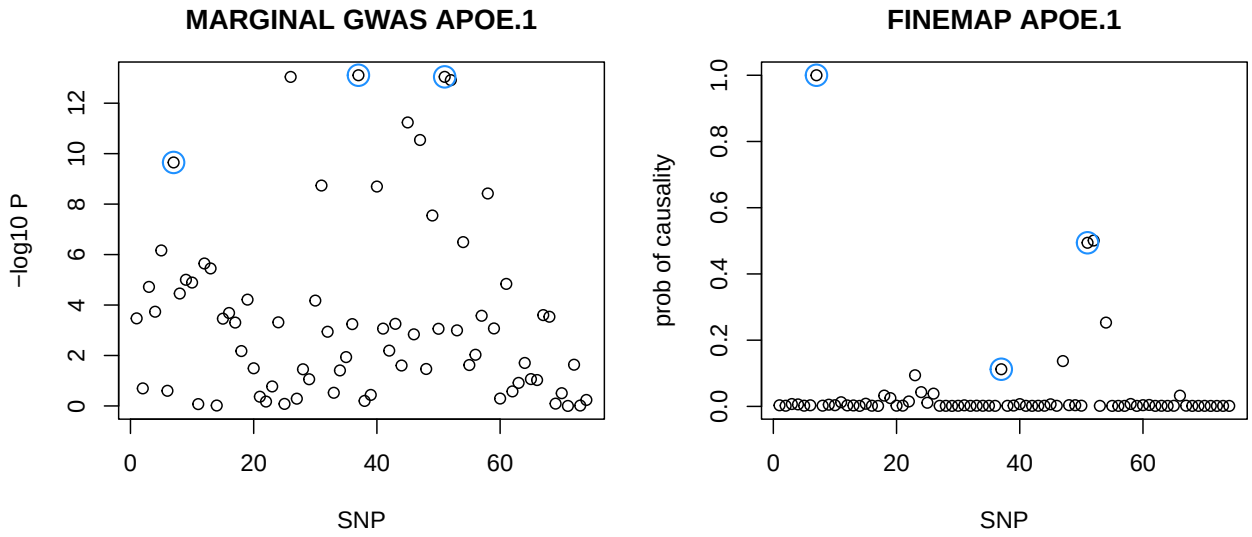
```
##    index      prob      mean        sd mean_incl   sd_incl
## 1      7 0.9999990 0.3249470 0.0703492  0.324947 0.0703488
## 2     52 0.5009420 0.1634080 0.1706030  0.326202 0.0706945
## 3     51 0.4941250 0.1599220 0.1691680  0.323647 0.0701917
## 4     54 0.2531170 0.0816844 0.1440640  0.322714 0.0648974
## 5     47 0.1368880 0.0442355 0.1136660  0.323150 0.0652116
## 6     37 0.1122280 0.0359365 0.1032020  0.320211 0.0622381
## 7     23 0.0940849 0.0339350 0.1075790  0.360685 0.0717884
## 8     24 0.0430625 0.0149818 0.0716016  0.347909 0.0567978
## 9     26 0.0385936 0.0123219 0.0620231  0.319272 0.0409315
## 10    18 0.0326849 0.0116583 0.0643032  0.356689 0.0586465
```

Here `prob` is the PIP, i.e., probability that this is one of the causal variants, `mean` is the estimated allelic causal effect $\lambda$, averaged over all possible causal sets and `mean_incl` is the estimate of $\lambda$ over those causal sets in which this SNP is included.

The three simulated causals were 7, 37 and 51. The first is labelled as a definite causal with PIP ~ 100%. The variant 51 cannot be as perfectly singled out from the other correlated variant 52, but gets PIP of ~ 50%. The variant 37 is sharing its effect with several other correlated variants and has a PIP of only 11%. The causal effects in `mean_incl` seem estimating the true value of 0.3 given their uncertainty as measured by posterior SD.

Let's plot the marginal GWAS P-value and FINEMAP's PIPs, and highlight the true causal variants.

```
par(mfrow = c(1,2))
par(mar = c(4.5,4.5,3,1))
plot(-log10(res.1[3,]), xlab = "SNP", ylab = "-log10 P", main = "MARGINAL GWAS APOE.1")
points(c.ind.1, -log10(res.1[3,c.ind.1]), cex = 2, lwd = 1.4, col = "dodgerblue")
plot(fm.1.snp$index, fm.1.snp$prob, xlab = "SNP", ylab = "prob of causality",main = "FINEMAP APOE.1")
i = which(fm.1.snp$index %in% c.ind.1)
points(fm.1.snp[i,"index"], fm.1.snp[i,"prob"], cex = 2, lwd = 1.4, col = "dodgerblue")
```




FINEMAPping has given a clearer picture here as there are less variants with interesting PIPs after fine-mapping than there are in the marginal P-values. Note how FINEMAP is most certain that variant 7 is a causal variant whereas the marginal P-value was smaller for variant 37 than variant 7. A likely reason for this difference is that 37 has more LD-friends with which it is forced to share some probability of being causal, whereas 7 is less correlated with others, and hence more clearly a causal itself. Such information cannot be extracted from the marginal P-values.

The results in .snp file are SNP-wise summaries over all candidate causal configurations that FINEMAP has evaluated. By default .config file reports top 50,000 of these configs. Let's show a few top ones, with posterior probabilities and estimated proportion of variance explained by the config ($h^2$, heritability).

```
fm.1.config = read.table(paste0(fm.path,"1.config"), as.is = T, header = T)
fm.1.config[1:10,c(2:4,8:9)]
```

```
##     config      prob log10bf        h2          h2_0.95CI
## 1     7,52 0.1566880 25.6341 0.0945202 0.0655098,0.130518
## 2  7,51,54 0.0954011 27.2819 0.0960810 0.0635471,0.130148
## 3     7,51 0.0678818 25.2708 0.0936830  0.0613809,0.12744
## 4  7,52,54 0.0580591 27.0662 0.0966661 0.0686048,0.133019
## 5  7,47,51 0.0560398 27.0509 0.1068330  0.0735969,0.14681
## 6  7,47,52 0.0396423 26.9005 0.1061010 0.0727301,0.140631
## 7  7,37,51 0.0378142 26.8800 0.1072280 0.0745664,0.148666
## 8  7,23,51 0.0357141 26.8552 0.1005670   0.0695624,0.1359
## 9  7,37,52 0.0332142 26.8237 0.1067970  0.0749096,0.14436
## 10 7,23,52 0.0306849 26.7893 0.1003580 0.0718231,0.135253
```

Let's also see the credible sets given by FINEMAP.

```
fm.1.cred = read.table(paste0(fm.path,"1.cred3"), as.is = T, header = T)
fm.1.cred[,-1]
```

```
##    cred1    prob1 cred2    prob2 cred3      prob3
## 1      7 0.999999    51 0.621653    54 0.31259300
## 2     NA       NA    52 0.378324    47 0.18362100
## 3     NA       NA    NA       NA    37 0.12390300
## 4     NA       NA    NA       NA    23 0.11702100
## 5     NA       NA    NA       NA    26 0.04635470
## 6     NA       NA    NA       NA    24 0.04327490
## 7     NA       NA    NA       NA    18 0.03420410
## 8     NA       NA    NA       NA    19 0.02162160
## 9     NA       NA    NA       NA    22 0.01868640
## 10    NA       NA    NA       NA    66 0.01666490
## 11    NA       NA    NA       NA    25 0.00935408
## 12    NA       NA    NA       NA    40 0.00696822
## 13    NA       NA    NA       NA    45 0.00555869
## 14    NA       NA    NA       NA     3 0.00500067
## 15    NA       NA    NA       NA    15 0.00482753
## 16    NA       NA    NA       NA    11 0.00332905
```

For credible sets (with 3 causal variants), FINEMAP takes the top causal 3-SNP config (here 7,51,54), and then asks, for each variant in the top configuration, which are the other candidate variants that could possibly replace the top variant in this causal configuration.

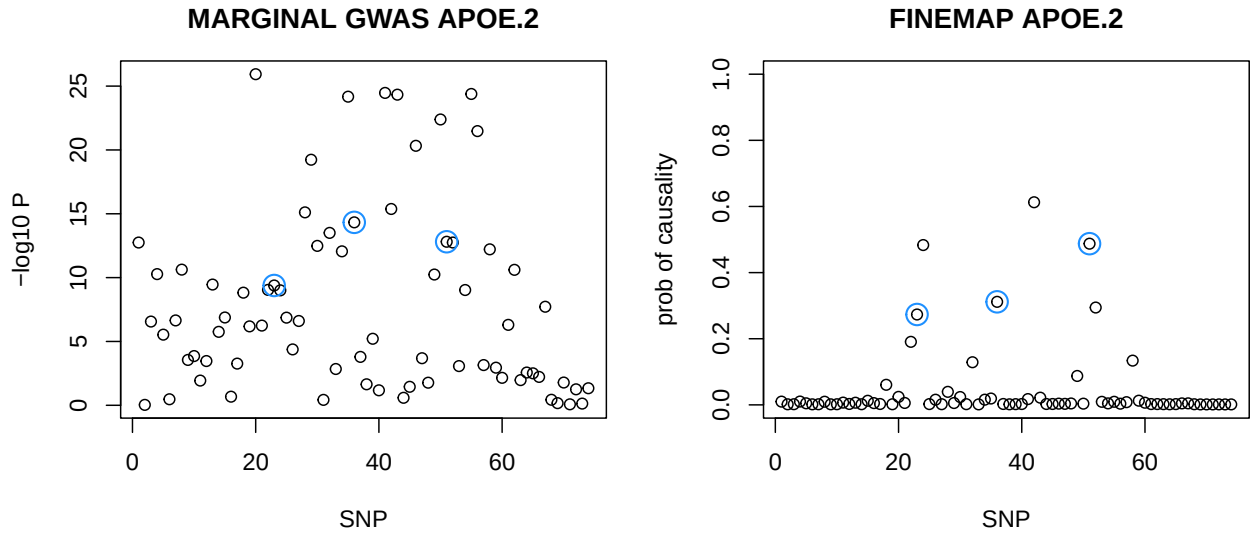
For example, column `cred_1` shows that the credible set $A_1 = \{7\}$ includes only SNP 7 as it already covers 0.99999 of the posterior probability of being in the causal set with 51 and 54. In other words, the sum of probabilities of any other variant replacing 7 in this causal set, that additionally contains SNPs 51 and 54, is $< 0.00001$.

For the second credible set, $A_2 = \{51, 52\}$, the correct causal variant 51 is the best candidate but there is also another candidate, 52, that has a slightly smaller probability than 51 to be the second causal variant.

For the third credible set, $A_3$, the picture is less clear and we need 16 variants before we cover 95% of the probability of including a causal variant in $A_3$. The known true causal variant, 37, is the third variant in the list.

A credible set collects the most likely causal variants and is a place to search for known functional variants that could be responsible of the association signal. If the credible set is small, it can provide candidate variants to be taken further to experimental studies that could then identify the causal variant among the candidates via its biological function. The size of the credible set depends mainly on the GWAS sample size and the LD-structure in such a way that a large sample size and lower levels of LD decrease the size of the credible sets.

**FINEMAP on APOE.2** On APOE.2 data causals were 23, 36 and 51. but due to LD, there were many other variants that had much lower P-values than the true causals. Conditional analysis chose 20 and 66, which was a clearly worse fit than the true causal set. Let's see FINEMAP's results. (Suppressing plotting commands from document.)




We see that FINEMAP gives PIPs between 20% and 50% to the true causals, and that there are two non-causals getting even higher PIPs. Even if the true causals are not having the highest PIPs here, the FINEMAP results still seem more robust for capturing the true causal variants than the marginal P-values.

Let's check the top 15 configs.

```
##       config      prob log10bf       h2          h2_0.95CI
## 1   24,42,51 0.0572840 32.2767 0.123297 0.0848914,0.163701
## 2   24,42,52 0.0491410 32.2101 0.123032 0.0877404,0.165497
## 3   23,42,51 0.0405076 32.1262 0.123364 0.0892675,0.162275
## 4   24,42,58 0.0397335 32.1178 0.122725  0.0888049,0.16374
## 5   22,42,51 0.0369072 32.0858 0.123093 0.0902163,0.160739
## 6   24,36,51 0.0353639 32.0672 0.122070 0.0865942,0.160689
## 7   24,36,52 0.0303078 32.0002 0.121804 0.0866079,0.165373
## 8   24,42,49 0.0275196 31.9583 0.121163  0.086644,0.160704
## 9   24,36,58 0.0227966 31.8765 0.121409 0.0858296,0.157195
## 10 23,42,52 0.0215781 31.8527 0.122486 0.0879697,0.161271
## 11 23,32,51 0.0215098 31.8513 0.121468 0.0890243,0.163025
## 12 23,36,51 0.0213135 31.8473 0.121976 0.0871476,0.159492
## 13 22,42,52 0.0187538 31.7917 0.122161 0.0864746,0.160347
## 14 24,36,49 0.0181203 31.7768 0.119969 0.0858125,0.157738
## 15 22,36,51 0.0168744 31.7459 0.121530 0.0867643,0.160187
```

The top FINEMAP config is (24,42,51) whereas the true causal config (23,36,51) has rank 12. Let's compare statistical evidence between the top FINEMAP config, the true causal config and the conditional analysis config (20,66).

```
lm.fm = lm(y.2 ~ X[,c(24,42,51)])
lm.causal = lm(y.2 ~ X[,c(23,36,51)])
lm.step = lm(y.2 ~ X[,c(20,66)])
data.frame(finemap = BIC(lm.fm), true_causal = BIC(lm.causal), stepwise = BIC(lm.step))
```

```
##    finemap true_causal stepwise
## 1 2716.861    2718.804 2742.155
```

Here the top config from FINEMAP happens to be a slightly better description of the data than the true causal config, which is a result of statistical sampling effects in a finite data set. We also see that the

conditional analysis has failed badly here compared to FINEMAP in terms of explaining the data (26 units difference in BIC).

We conclude that FINEMAP has made an efficient search through the space of configurations and identified a config with more statistical evidence than the true causal config. Given the data available, we cannot expect to get better fine-mapping results based on statistical evidence alone.

**7.5.1 Fine-mapping assuming one causal variant** Sometimes we do not have access to the original GWAS data and therefore we cannot accurately estimate LD that corresponds to the GWAS summary statistics. This means that we cannot do complete fine-mapping with FINEMAP or other similar methods that require LD information. In these cases, it is still possible to compute, for each variant in the region, the posterior probability of causality under the assumption that **there is at most one causal variant in the region and it is included in the study**. The approach was introduced by Maller et al. 2012, together with the idea of credible sets.

Let's mark by $H_l, l = 1, \ldots, p$ the hypothesis that says $l$ is the only causal variant and let $H_0$ be the null hypothesis that says there is no causal variant in the region. Mark by $H_C = H_1 \cup \ldots \cup H_p$ the hypothesis that says that there is exactly one causal variant in the region. We assume that, *a priori*, each variant is equally likely to be the causal variant, i.e., $\Pr(H_l \mid H_C) = \frac{1}{p}$.

It follows that the regional Bayes factor is

$$\mathrm{BF}_{\mathrm{reg}} = \frac{\Pr(\mathrm{Data} \mid H_C)}{\Pr(\mathrm{Data} \mid H_0)} = \frac{\sum_{l=1}^{p} \Pr(\mathrm{Data} \mid H_l) \Pr(H_l \mid H_C)}{\Pr(\mathrm{Data} \mid H_0)} = \frac{\sum_{l=1}^{p} \frac{1}{p} \Pr(\mathrm{Data} \mid H_l)}{\Pr(\mathrm{Data} \mid H_0)} = \frac{1}{p} \sum_{l=1}^{p} \mathrm{BF}_l,$$

where $\mathrm{BF}_l$ is the single variant Bayes factor that does not depend on other variants or the LD structure of the region. In practice, it can be computed as Approximate Bayes Factor (ABF) from GWAS summary statistics, as was explained in Section 4 of the course material.

The posterior probability for variant $l$ is then, according to Bayes formula,

$$\Pr(H_l \mid \mathrm{Data}, H_C) = \frac{\Pr(H_l \mid H_C) \Pr(\mathrm{Data} \mid H_l)}{\Pr(\mathrm{Data} \mid H_C)} = \frac{1}{p} \frac{\frac{\Pr(\mathrm{Data} \mid H_l)}{\Pr(\mathrm{Data} \mid H_0)}}{\frac{\Pr(\mathrm{Data} \mid H_C)}{\Pr(\mathrm{Data} \mid H_0)}} = \frac{1}{p} \frac{\mathrm{BF}_l}{\mathrm{BF}_{\mathrm{reg}}} = \frac{\mathrm{BF}_l}{\sum_{k=1}^{p} \mathrm{BF}_k} \propto \mathrm{BF}_l.$$

Thus, the posterior probability that SNP $l$ is the single causal variant in the region, under the assumption that there is exactly one causal variant, is proportional to the single-SNP Bayes factor of SNP $l$.

In practice, we get the posterior probabilities by computing ABF for each variant and then normalizing them to sum to one. This derivation is independent of LD in the region because, under the assumption of only a single causal SNP, the marginal effect at the causal SNP is the causal effect itself, independent of the LD with other variants. With strongly associated but also highly correlated SNPs, we are left with the result that any of them could be the causal variant. This will manifest through high single-SNP and regional BFs, but with the posterior probability distributed fairly evenly across the highly correlated SNPs.

We have assumed that the causal SNP is included in the study. When this is not true, the above approach is still applicable if there is a good surrogate SNP for the causal effect. In the presence of multiple causal SNPs, this approach is no longer optimal. It will tend to pick out the SNP with the best marginal effect, which may or may not be one of the causal SNPs.

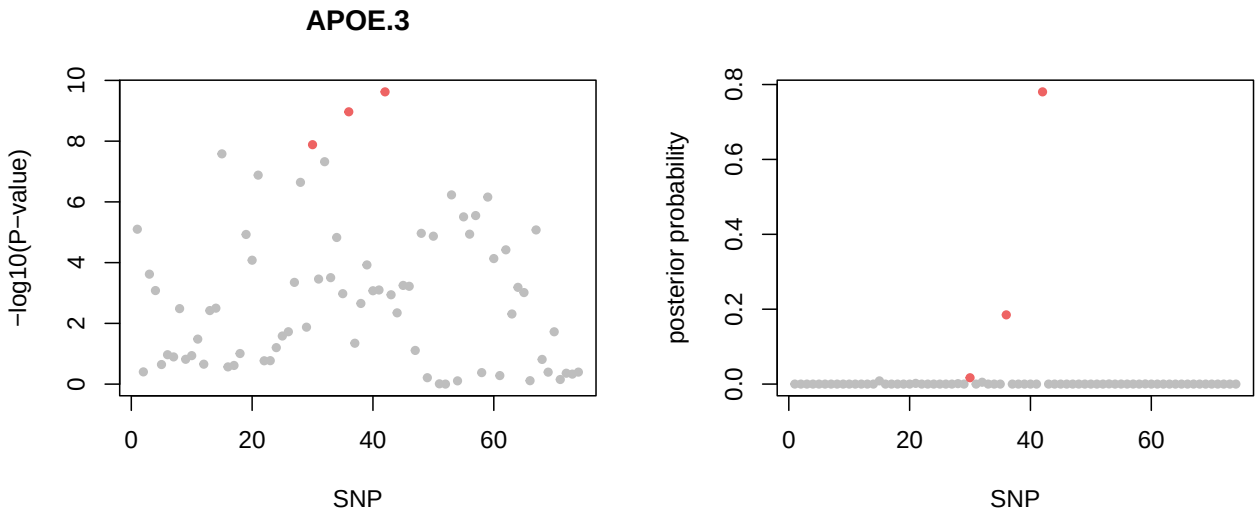
**Example 7.6: APOE.3 data.** Let's make yet another phenotype data set for the *APOE* region genotype data that we currently have in `X` matrix. Now we include only one causal variant, SNP 42.

```
set.seed(6)
c.ind = c(42)
lambda = rep(0, p)
lambda[c.ind] = 0.3
y.3 = X %*% lambda + rnorm(n, 0, sqrt(1 - var(X %*% lambda)) )
```

```r
#Data has been generated. Now run a GWAS one variant at a time.
res.3 = apply(X, 2, function(x){summary(lm(y.3 ~ x))$coeff[2, c(1,2,4)]})
#Compute single-SNP ABFs using the GWAS summary statistics
tau = 0.2 #prior SD of effect size
b.est = res.3[1,]
se = res.3[2,]
log.abf = dnorm(b.est, 0 , sqrt(tau^2 + se^2), log = T) - dnorm(b.est, 0, se, log = T)
abf = exp(log.abf)
#Normalize ABFs to get PIPs of the variants assuming that there is exactly one causal variant
posterior = abf/sum(abf)
#Let's see which variants have probability > 1%
ind = which(posterior > 0.01)
ind = ind[order(posterior[ind], decreasing = T)] #Order them from largest to smallest
data.frame(SNP = ind, prob = posterior[ind], Pvalue = res.3[3,ind])
```

```
##     SNP       prob       Pvalue
## V42  42 0.78045144 2.384397e-10
## V36  36 0.18488980 1.081044e-09
## V30  30 0.01688366 1.308594e-08
```

```r
#Let's plot the -log10 P-values and posterior probabilities
par(mfrow = c(1,2))
cols = rep("gray", p)
cols[ind] = "indianred2"
plot(-log10(res.3[3,]), xlab = "SNP", ylab = "-log10(P-value)", main = "APOE.3",
     cex = 0.7, col = cols, pch = 19)
plot(posterior, xlab = "SNP", ylab = "posterior probability", main = "",
     cex = 0.7, col = cols, pch = 19)
```




We see that the 3 smallest P-values also have the largest posterior probabilities of being causal. The SNP 42 (78%) has about 4 times higher probability than SNP 36 (18%) and 50 times higher probability than SNP 30 (1.6%) of being the causal variant in the region (when we assume that there is exactly one causal variant).

To determine the 95% credible set, we would include the SNPs with the highest posteriors until their cumulative sum is $> 95\%$. Here this set would contain only SNPs 42 and 36 as their sum is already $78\% + 18\% = 96\%$.

If we would have several conditionally independent association signals in the region, as returned, for example, by the stepwise forward search, then we could fine-map each of them separately. For this, we would compute the association statistics (beta and SE) by including the SNPs representing the other signals in the region as covariates, and then carry out the fine-mapping for each signal separately as above by assuming that there is only one causal variant left after conditioning on the other signals in the region. (In Exercise 5.2 we do that for one signal conditioned on two other signals.)