Genome wide assoc and Individual inbreeding

Jerome Goudet

2015-07-15

Contents

Inbreeding, relatedness and k coefficients

Let us start with a function for calculating individual inbreeding coefficient using the method of moments (slides 207-208)

```
ind.inb<-function(data){
ni<-dim(data)[1]
nl<-dim(data)[2]-1
ndat<-getal.b(data)
pfreq<-pop.freq(cbind(rep(1,ni),data[,-1]))
ifreq<-pop.freq(cbind(1:ni,data[,-1]))
num<-0.0
den<-0.0
for (il in 1:nl){
num<-num+apply((sweep(ifreq[[il]]*2,1,2*pfreq[[il]]))^2,2,sum)
den<-den+sum((2*pfreq[[il]]*(1.0-pfreq[[il]])))
}
inb<-num/den-1  #F3 on slide 208
return(inb)
}
```

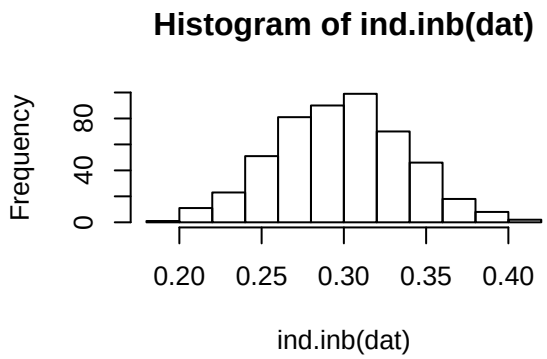
As an exercice, try programming $\hat{F}_1, \hat{F}_2$ and $\hat{F}_4$ (slides 207)

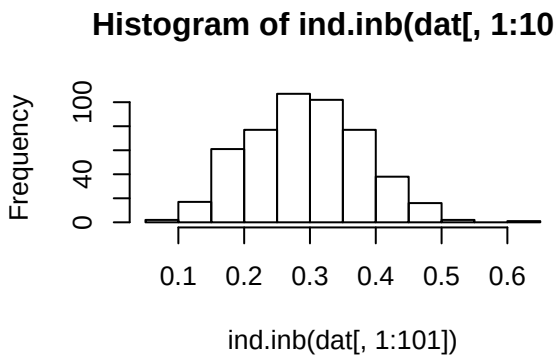
Let's now simulate some data:

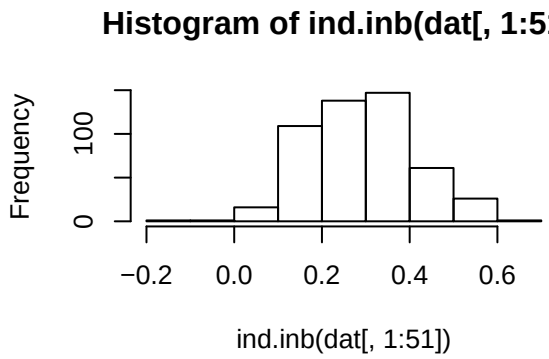
```
library(hierfstat)
dat<-sim.genot(nbloc=500,nbal=20,size=500,nbpop=1,f=0.3)
```

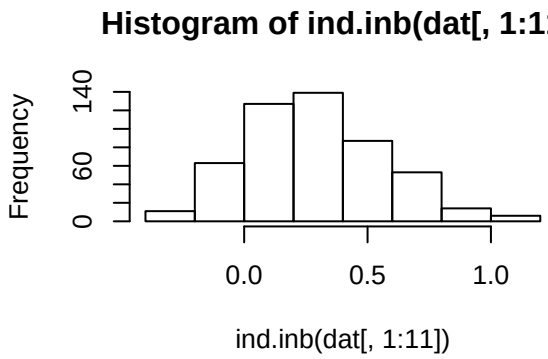
and plot the individual inbreeding coeff:

```
par(mfrow=c(2,2))
hist(ind.inb(dat)) #all 500 loci
hist(ind.inb(dat[,1:101])) #only 100
hist(ind.inb(dat[,1:51])) #50
hist(ind.inb(dat[,1:11])) #10
```










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

We move now to amd data set and will do some calculation with it:

```
amd <- read.table("AMD.txt", header=T)
control1<-amd[amd$chromosome==1,101:150]
ncontrols<-dim(control1)[2]
co1<-apply(control1,1,fun<-function(x) sum(x==1))
co2<-apply(control1,1,fun<-function(x) sum(x==2))
co3<-apply(control1,1,fun<-function(x) sum(x==3))
mco<-apply(control1,1,fun<-function(x) sum(x==0))
control.counts<-cbind(co1,co2,co3)
pco1<-(co3*2+co2)/2/(ncontrols-mco)
ncontrol1<-as.matrix(control1)-1
x<-which(ncontrol1==-1)
ncontrol1[x]<-NA
head(ncontrol1[,1:10])
```

```
##   X16151536 X18954665 X19324516 X19364338 X20928240 X21775277 X23445875
## 3         2         2         2         2         2         2         2
## 4         2         2         2         2         2         2         2
## 5         0         0         0         0         0         0         0
## 6         2         0         0         0         1         1         0
## 7         2         0         0         0         1         1         0
## 8         0         2         2         2         1         1         2
```

```
##   X26477587 X26497242 X29767105
## 3         2         2         2
## 4         2         2         2
## 5         0         0         0
## 6         1         0         0
## 7         1         0         0
## 8         1         2         2
```

the inbreeding coeffs of these individuals:

```
(inb.coeff<-apply(ncontrol1,2,
    function(x){
     nnas<-which(!is.na(x))
     sum((x[nnas]-2*pco1[nnas])^2)/
     sum(2*pco1[nnas]*(1-pco1[nnas]))-1
     }))
```

```
##     X16151536     X18954665     X19324516     X19364338     X20928240
##  0.0002232592 -0.0075810213 -0.0084150111 -0.0221856438 -0.0397993937
##     X21775277     X23445875     X26477587     X26497242     X29767105
##  0.0255739215 -0.0128005997  0.0246130613 -0.0307963954  0.0484263294
##     X33573719     X36676484     X37776428     X39852279     X43629329
## -0.0158381226 -0.0320619952 -0.0571629816 -0.0264417938  0.0471656415
##     X47724985     X48659034     X49148634     X53133899     X54851882
## -0.0054446386  0.0065262543  0.0847667162  0.0142727279 -0.0235678344
##     X63160264     X65602476     X69972363     X70284578     X70309242
## -0.0043743354 -0.0460886936 -0.0365960497 -0.0214315901 -0.0564184472
##     X70855215     X75062383     X78283159     X78726432     X80248015
## -0.0417973198  0.0301677113 -0.0385943536 -0.0109110389 -0.0059256415
##     X81097374     X81563012     X81891266     X84094938     X84563724
##  0.0211427147  0.0095329887 -0.0733502964  0.0027408454 -0.0394955789
##     X85560309     X87913336     X97877008      M1145794      M2122799
## -0.0208129167  0.0498428935 -0.0356586215  0.0550758178 -0.0194077491
##      M3133062      M3746788      M5300976      M5312787      M5393685
## -0.0491632756 -0.0157492454  0.0059719257  0.0160024693 -0.0098719249
##      M5623682      M6422757      M7268182      M8081308      M9220221
## -0.0237048187 -0.0041821915 -0.0035167936 -0.0208982932  0.0074536865
```

```
range(inb.coeff)
```

```
## [1] -0.07335030  0.08476672
```

A function for the coancestry coeff (slide 219)

```
coancestry.coeff<-function(x,y,genos=ncontrol1,af=pco1){
nnas<-which(!is.na(genos[,x]) & !is.na(genos[,y]))
num<-sum((genos[nnas,x]-2*af[nnas])
         *(genos[nnas,y]-2*af[nnas]))
den<-sum(2*af[nnas]*(1-af[nnas]))
coa<-num/den/2
if(x==y) coa<-2*coa-1 #inbreeding coefficient
return(coa)
}
```

and for the k_0, k_1, k_2 coeffs (see slides 214-217)

```
k.coeff<-function(x,y,genos=ncontrol1,af=pco1){
nnas<-which(!is.na(genos[,x]) & !is.na(genos[,y]))
N0<-sum((genos[nnas,x]==2 & genos[nnas,y]==0) | (genos[nnas,x]==0 & genos[nnas,y]==2))
N2<-sum(genos[nnas,x]==genos[nnas,y])
N1<-length(nnas)-N0-N2
naf<-af[nnas]
k0<-N0/2/sum(naf^2*(1-naf)^2)
k1<-(N1-4*k0*sum(naf*(1-naf)*(naf^2+(1-naf)^2)))/2/sum(naf*(1-naf))
k2<-(N2-k0*sum((naf^2+(1-naf)^2)^2)-k1*sum(naf^3+naf*(1-naf)+(1-naf)^3))/length(naf)
#or
#k2<-1-k0-k1
coan<-.25*k1+.5*k2
return(c(k0,k1,k2,coan))
}
```

Now these control individuals are not related. Thus plotting k0 vs k1 (slide 218) won't be very informative. This kind of plot is interesting when we have relatives in the dataset. Let's create some dummy individuals from the AMD control individuals, and estimate their relatedness and k coeffs. Code below for these crosses is a bit more involved, but it will be a good exercice for you to wlak through it.

```
######################generate genotypes from given crosses####################
##this is to simulate PO, HS, FS
#use AMD control as parents
gam<-array(dim=c(dim(ncontrol1),2))

gam1<-matrix(rep(NA,prod(dim(ncontrol1))),ncol=dim(ncontrol1)[2])
nnas<-!is.na(ncontrol1)
gam1[ncontrol1==2]<-1
gam1[ncontrol1==1]<-sample(0:1,size=1)
gam1[ncontrol1==0]<-0
gam2<-gam1
gam2[nnas][ncontrol1[nnas]==1]<-1-gam1[nnas][ncontrol1[nnas]==1]

gam[,,1]<-gam1
gam[,,2]<-gam2


draw.cross<-function(p1=1,p2=2,noff=2,gam=gam){
#p1 is id of first parent, p2 is id of second parent
#noff is number of offsprings to draw from cross
#gam [nloc,2 or more parents,2 chromo] contains the parental chromosomes
nloc<-dim(gam)[1]
nnas<-which(!is.na(gam[,p1,1]) & !is.na(gam[,p2,1]))
ogenos<-matrix(rep(NA,nloc*noff),ncol=noff)
for (io in 1:noff){
ogenos[nnas,io]<-sapply(nnas,fun<-function(x)
  gam[x,p1,sample(1:2,size=1)]+
    gam[x,p2,sample(1:2,size=1)])
}
return(ogenos)
}
```

```
draw.cross1<-function(p1=1,p2=2,noff=2,genos=ncontrol1){
#p1 is id of first parent, p2 is id of second parent
#noff is number of offsprings to draw from cross
#genos [nloc*2 or more] contains parental genotypes
ogeno<-function(pg){
#return offspring genotype
if (is.na(pg[1]) | is.na(pg[2])) return(NA) else{
if (pg[1]==0 & pg[2]==0) return(0)
if ((pg[1]==0 & pg[2]==1) | (pg[1]==1 & pg[2]==0)) return(sample(0:1,size=1))
if ((pg[1]==0 & pg[2]==2) | (pg[1]==2 & pg[2]==0)) return(1)
if (pg[1]==1 & pg[2]==1) return(sample(0:2, size=1, prob=c(.25,.5,.25)))
if ((pg[1]==1 & pg[2]==2) | (pg[1]==2 & pg[2]==1)) return(sample(1:2,size=1))
if (pg[1]==2 & pg[2]==2) return(2)
}
}
nloc<-dim(genos)[1]
nnas<-which(!is.na(genos[,p1]) & !is.na(genos[,p2]))
off.geno<-matrix(rep(NA,nloc*noff),ncol=noff)
np<-ncontrol1[,c(p1,p2)]
for (io in 1:noff)
off.geno[,io]<-unlist(apply(np,1,ogeno))
return(off.geno)
}
```

We now draws different types of offsprings, in order to create full sibs, half sibs, parent offspring etc...

```
noff<-5
o.12<-draw.cross(1,2,noff=noff,gam=gam)
o.13<-draw.cross(1,3,noff=noff,gam=gam)
o.14<-draw.cross(1,4,noff=noff,gam=gam)
o.23<-draw.cross(2,3,noff=noff,gam=gam)
o.24<-draw.cross(2,4,noff=noff,gam=gam)
o.34<-draw.cross(3,4,noff=noff,gam=gam)

o.55<-draw.cross(5,5,noff=noff,gam=gam) #selfing!
ohs<-draw.cross1(1,2,noff=noff,genos=
            cbind(o.13[,1],o.34[,1])) #HS mating


test.o<-cbind(o.12,o.13,o.14,o.23,o.24,o.34,
      ncontrol1[,1:7],o.55,ohs)
```

Now the calculations:

```
#set matrices to store k coeffs
noff<-dim(test.o)[2]
coan.off<-matrix(numeric(noff^2),ncol=noff)
k1.off<-matrix(numeric(noff^2),ncol=noff)
k2.off<-matrix(numeric(noff^2),ncol=noff)
k0.off<-matrix(numeric(noff^2),ncol=noff)
coanb.off<-matrix(numeric(noff^2),ncol=noff)


(nl<-length(pco1))
```

```
## [1] 9798
```

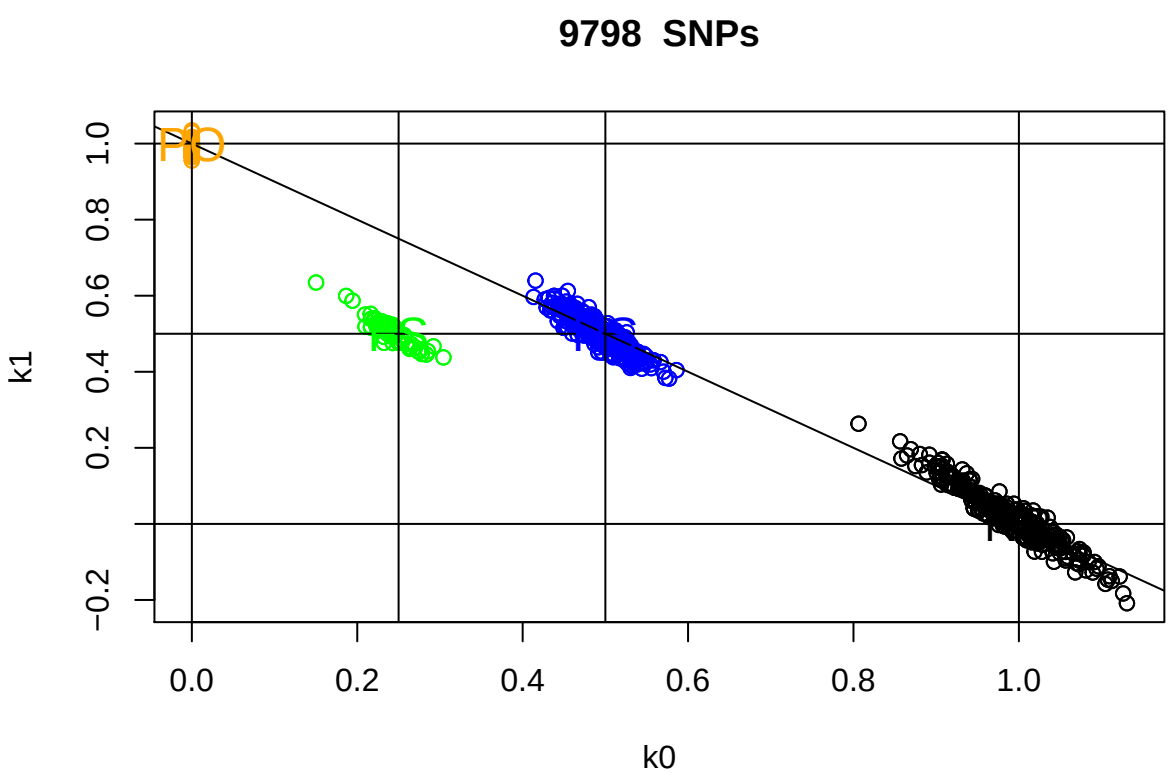
```
s<-1:nl

#loops over all pairs of individuals

for (i in 2:noff){
for (j in 1:(i-1)){
k.off<-k.coeff(i,j,genos=test.o[s,],af=pco1[s])
k0.off[i,j]<-k.off[1]
k1.off[i,j]<-k.off[2]
k2.off[i,j]<-k.off[3]
coanb.off[i,j]<-k.off[4]
coanb.off[j,i]<-coanb.off[i,j]
k0.off[j,i]<-k0.off[i,j]
k1.off[j,i]<-k1.off[i,j]
k2.off[j,i]<-k2.off[i,j]
}
}
```

Prepare for plotting with different colors:

```
x<-c(1:37) #only non inbreds inds
mcol<-matrix(character(prod(dim(k0.off[x,x]))),ncol=37)
mcol[k0.off[x,x]>0.8]<-"black"
mcol[k0.off[x,x]<0.8 & k0.off[x,x]>0.35]<-"blue"
mcol[k0.off[x,x]<0.35 & k0.off[x,x]>0.1]<-"green"
mcol[k0.off[x,x]<0.05 & k1.off[x,x]>.8]<-"orange"
mcol[k0.off[x,x]<.05 & k1.off[x,x]<.05]<-"white"
```

And now for the plotting

```
plot(k0.off[x,x],k1.off[x,x],xlab="k0",ylab="k1",col=mcol)
abline(c(1,-1))
abline(h=c(0,0.5,1.0))
abline(v=c(0,0.25,0.5,1.0))
text(0.25,0.5,"FS",col="green",cex=1.5)
text(0.5,0.5,"HS",col="blue",cex=1.5)
text(0,1,"PO",col="orange",cex=1.5)
text(1,0,"NR",col="black",cex=1.5)
title(paste(nl," SNPs"))
```



now using only 1000 SNPs

```
(nl<-1000)
```

```
## [1] 1000
```

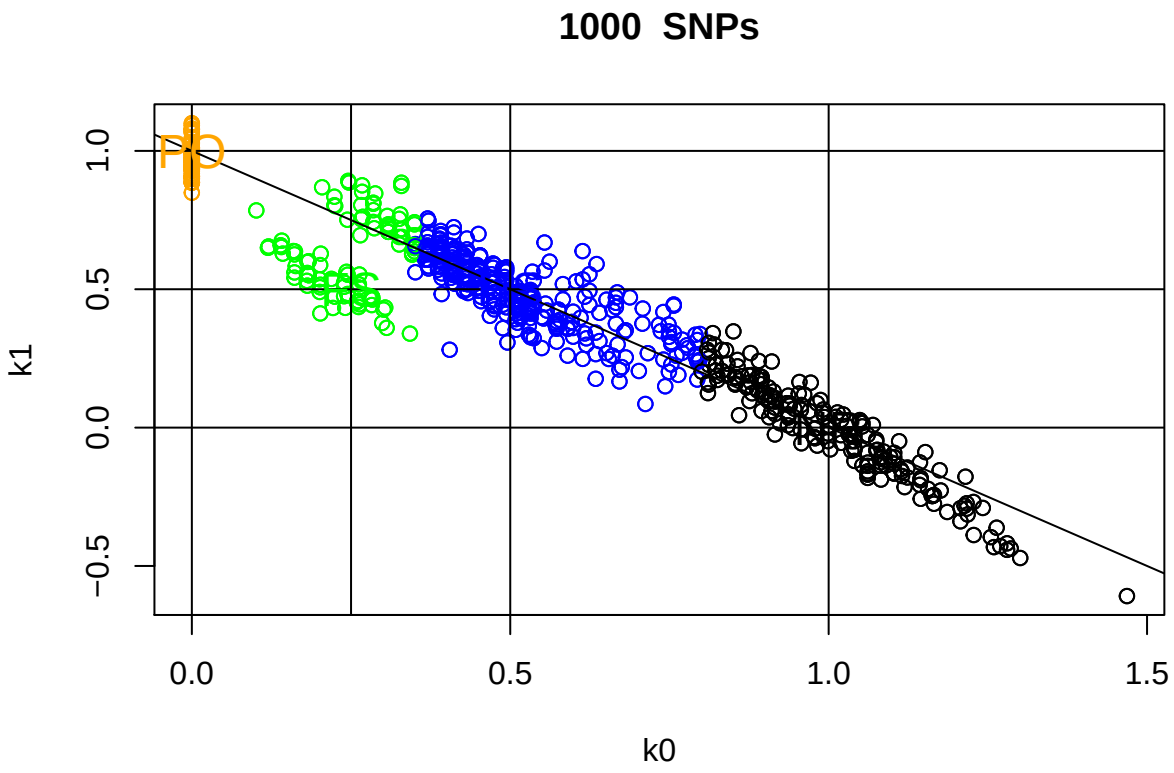
```
s<-1:nl

#loops over all pairs of individuals

for (i in 2:noff){
for (j in 1:(i-1)){
k.off<-k.coeff(i,j,genos=test.o[s,],af=pco1[s])
k0.off[i,j]<-k.off[1]
k1.off[i,j]<-k.off[2]
k2.off[i,j]<-k.off[3]
coanb.off[i,j]<-k.off[4]
coanb.off[j,i]<-coanb.off[i,j]
k0.off[j,i]<-k0.off[i,j]
k1.off[j,i]<-k1.off[i,j]
k2.off[j,i]<-k2.off[i,j]
}
}
```

Prepare for plotting with different colors:

```
x<-c(1:37) #only non inbreds inds
mcol<-matrix(character(prod(dim(k0.off[x,x]))),ncol=37)
mcol[k0.off[x,x]>0.8]<-"black"
mcol[k0.off[x,x]<0.8 & k0.off[x,x]>0.35]<-"blue"
mcol[k0.off[x,x]<0.35 & k0.off[x,x]>0.1]<-"green"
mcol[k0.off[x,x]<0.05 & k1.off[x,x]>.8]<-"orange"
mcol[mcol==""]<-"white"
```

And now for the plotting

```
plot(k0.off[x,x],k1.off[x,x],xlab="k0",ylab="k1",col=mcol)
abline(c(1,-1))
abline(h=c(0,0.5,1.0))
abline(v=c(0,0.25,0.5,1.0))
text(0.25,0.5,"FS",col="green",cex=1.5)
text(0.5,0.5,"HS",col="blue",cex=1.5)
text(0,1,"PO",col="orange",cex=1.5)
text(1,0,"NR",col="black",cex=1.5)
title(paste(nl," SNPs"))
```



Association mapping

```
library(hwde)
```

```
## Warning: package 'hwde' was built under R version 3.2.1
```

```
amd <- read.table("AMD.txt", header=T) #dataset need to be uncompressed first
amd1<-amd[which(amd$chromosome==1),] # extract snps from chrom 1 only
dim(amd1)
```

```
## [1] 9202  150
```

```
#cases are first 95 inds
case1<-amd1[,5:100]
ncases<-dim(case1)[2]
#control are the remainers
control1<-amd1[,101:150]
ncontrols<-dim(control1)[2]
#counts the diff genotypes for case
ca1<-apply(case1,1,fun<-function(x) sum(x==1))
ca2<-apply(case1,1,fun<-function(x) sum(x==2))
ca3<-apply(case1,1,fun<-function(x) sum(x==3))
mca<-apply(case1,1,fun<-function(x) sum(x==0))
case.counts<-cbind(ca1,ca2,ca3)
#hw test for cases
case.hwe<-apply(case.counts,1,fun<-function(x) hwexact(x[1],x[2],x[3]))
head(case.hwe)
```

```
##          3          4          5          6          7          8
## 0.02738382 1.00000000 1.00000000 0.04883257 0.25768409 0.01967943
```

```
#counts the different genotypes for control
co1<-apply(control1,1,fun<-function(x) sum(x==1))
co2<-apply(control1,1,fun<-function(x) sum(x==2))
co3<-apply(control1,1,fun<-function(x) sum(x==3))
mco<-apply(control1,1,fun<-function(x) sum(x==0))
control.counts<-cbind(co1,co2,co3)
#hw tests for control
control.hwe<-apply(control.counts,1,fun<-function(x) hwexact(x[1],x[2],x[3]))
head(control.hwe)
```

```
##         3         4         5         6         7         8
## 1.0000000 1.0000000 1.0000000 0.1877865 0.1877865 0.1877865
```

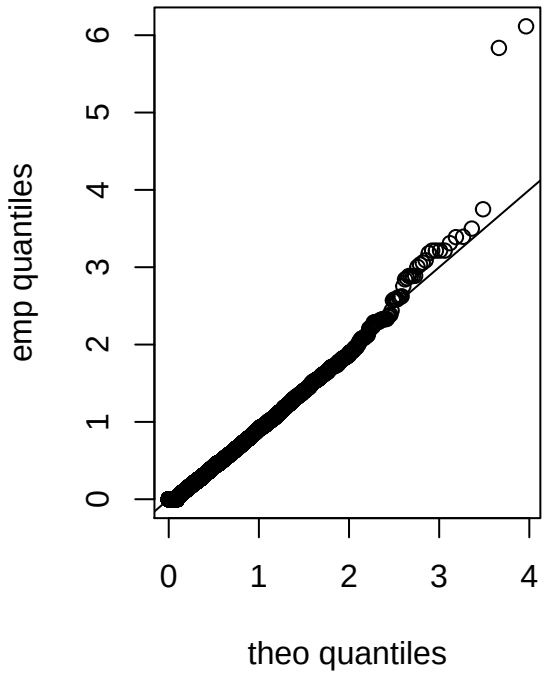
```
#why not testing them together by the way?
#maf in case
pca1<-(ca1*2+ca2)/2/(ncases-mca)
maf.ca1<-pca1
x<-which(maf.ca1>0.5)
maf.ca1[x]<-1-maf.ca1[x]
#maf in control
pco1<-(co1*2+co2)/2/(ncontrols-mco)
maf.co1<-pco1
x<-which(maf.co1>0.5)
maf.co1[x]<-1-maf.co1[x]
```

Do fisher exact test on maf counts of case vs control, e.g case-control test, p254 of handouts

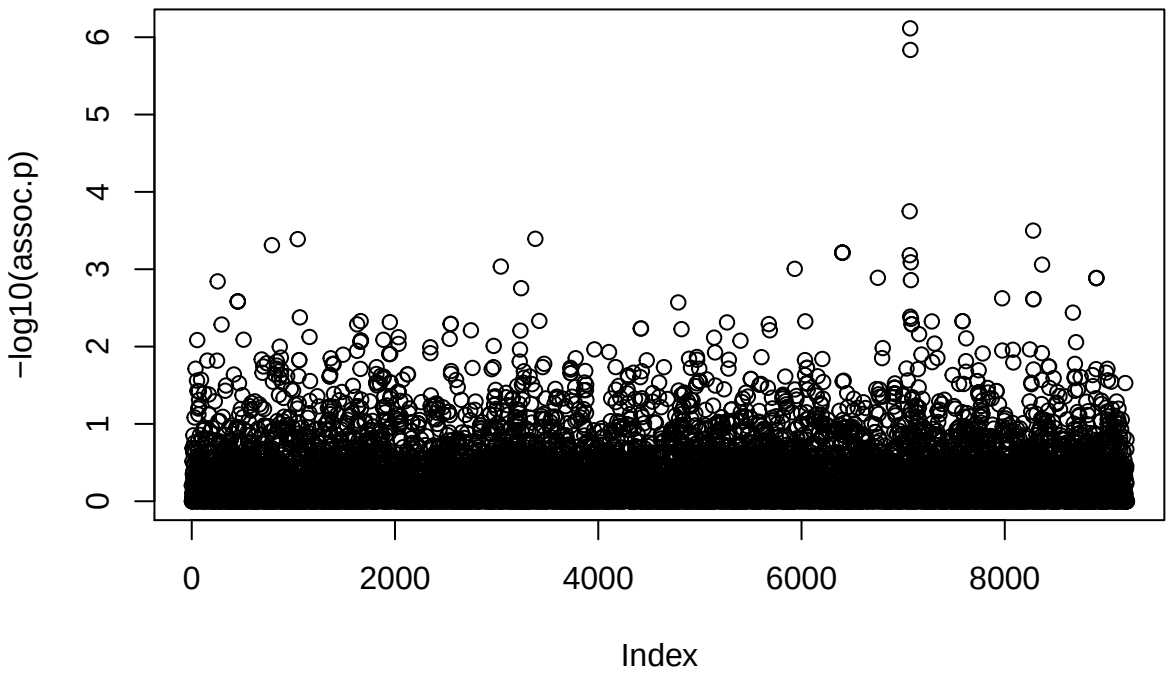
```
assoc.p<-apply(cbind(case.counts,control.counts),1,
               function(x) fisher.test(cbind(x[1:3],x[4:6]))$p.value)
```

And plot it

```
par(mfrow=c(1,2))
plot(-log10((length(assoc.p):1)/length(assoc.p)),
    -log10(sort(assoc.p,decreasing=T)),xlab="theo quantiles",
     ylab="emp quantiles");abline(c(0,1))
```




```
plot(-log10(assoc.p))
```



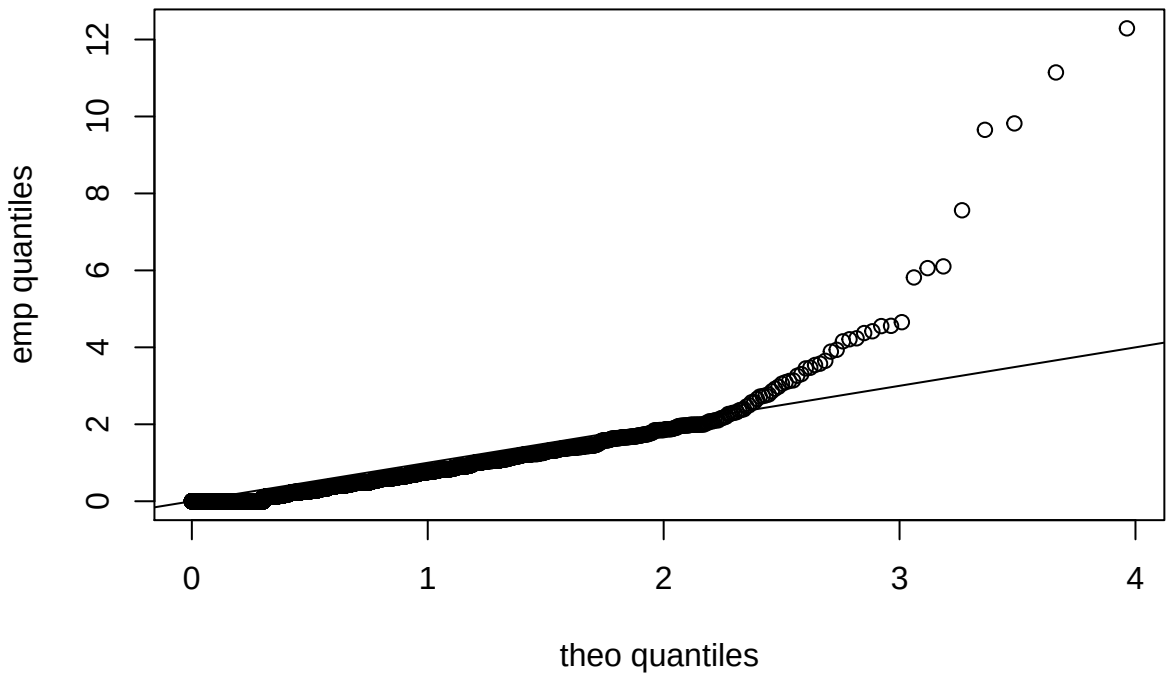

```
#set sig t-log10(5) arbitrarily
sig<-which(-log10(assoc.p)>5)
#Fig 1 B of Klein etal, 2005
amd1[sig,1:4]
```

```
##                SNP      rsID chromosome       pos
## 7075 SNP_A-1660027  rs380390          1 193989310
## 7077 SNP_A-1707207 rs1329428          1 193991069
```

```
cbind(case.counts[sig,],control.counts[sig,])
```

```
##      ca1 ca2 ca3 co1 co2 co3
## 7075  50  35  11   6  25  19
## 7077   2  24  68   5  29  14
```

```
#find out strong outliers
plot(-log10((length(control.hwe):1)/length(control.hwe)),
     -log10(sort(control.hwe,decreasing=T)),
     xlab="theo quantiles",ylab="emp quantiles")
abline(c(0,1))
```


emp quantiles
0
2
4
6
8
10
12
0
1
2
3
4
theo quantiles