Project 2

The following homework assignment should be submitted in writing no later than Monday, 8/2/2010. The work can be submitted in pairs

Estimating IBD for Half-Sibs: In class we considered the estimation of IBD in pedigrees, which are based on a pair of parents with their siblings. It was assumed that only the genotype of the siblings is available for the estimation. The *Forward* and *Backward* algorithms were applied in order to estimate the distribution of IBD at a locus given the genotypes of a collection of markers on the chromosome. Specifically, we assessed via simulation the efficiency of estimation for different densities of markers.

You are required to submit a report that summarizes the results of the analysis. The report should include an introduction section as well as sections on the method, the results and a discussion. In the discussion compare between the case of full-sibs that was discussed in class to your analysis of half-sibs.

Below we provide the code that was used in class for the estimation of IBD. Change the code to address data from affected half-sibs according to the following guiding questions. The relevant locations in the code are marked by the numbers of the questions:

1. Change the code of the example to have different fathers and the same mother. Notice that there are 6 unique sources of genetic material, rater than the 4 in the original code.

2. Change the function `ped.geno` to address the different number of genetic sources (6 instead of for). Notice that the range of values on only one of the two variables `v` and `par` changes.

3. Change the transition matrix. Observe that there are only two states in the hidden markov process. The probability of changing from one state to the other is $\phi = 1-\theta^2-(1-\theta)^2$, where θ is the probability of recombination.

4. Change the initial distribution of the two states of the markov change.

5. Modify the functions `forward`, `backward`, and `marginal`. Notice that there are 2 states to the hidden process instead of 3.

6. Modify the production of the array of probabilities of observed genotypes, given the IBD states. Notice that there are only two states of IBD with probabilities associated with the first two columns of the output of the function `geno.given.ibd`.

7. Modify the computation of the estimated IBD.

```
meiosis.link <- function(GF,GM,markers,qtl,inhe)
{
    n <- nrow(GF)
    GS <- GF
    loci <- sort(c(qtl, markers))
    rec.frac <- (1-exp(-0.02*diff(loci)))/2
    index <- 1:length(markers)
    from.GM <- inhe
    for (i in index[markers >= qtl])
    {
        rec <- rbinom(n,1,rec.frac[i])
        from.GM <- from.GM*(1-rec) + (1-from.GM)*rec
        GS[from.GM==1,i] <- GM[from.GM==1,i]
    }
    from.GM <- inhe
    for (i in rev(index[markers < qtl]))
    {
        rec <- rbinom(n,1,rec.frac[i])
        from.GM <- from.GM*(1-rec) + (1-from.GM)*rec
        GS[from.GM==1,i] <- GM[from.GM==1,i]
    }
    return(GS)
}

mating <- function(fa,mo,markers,qtl=markers[1],
    inhe.fa=rbinom(nrow(fa$pat),1,0.5),
    inhe.mo=rbinom(nrow(mo$pat),1,0.5))
{
    pat <- meiosis.link(fa$pat,fa$mat,markers,qtl,inhe.fa)
    mat <- meiosis.link(mo$pat,mo$mat,markers,qtl,inhe.mo)
    return(list(pat=pat, mat=mat))
}

# Question 1
n.ped <- 10
markers <- seq(0,140,by=20)
n.mark <- length(markers)
fa <- list(pat=matrix(1,n.ped,n.mark),
    mat=matrix(2,n.ped,n.mark))
mo <- list(pat=matrix(3,n.ped,n.mark),
    mat=matrix(4,n.ped,n.mark))
sib1 <- mating(fa,mo,markers)
sib2 <- mating(fa,mo,markers)
ibd <- (sib1$pat==sib2$pat)+
    (sib1$mat==sib2$mat)
ibd

genotype <- function(a1,a2,a3,a4,n.al=2)
{
    a.m <- pmin(a1,a2)
    a.M <- pmax(a1,a2)
    g1 <- a.M + (a.m-1)*(n.al-a.m/2)
    a.m <- pmin(a3,a4)
    a.M <- pmax(a3,a4)
    g2 <- a.M + (a.m-1)*(n.al-a.m/2)
    g <- g1 + (g2-1)*n.al*(n.al+1)/2
    return(g)
}

# Question 2
ped.geno <- function(sib1,sib2,f=rep(1/2,2))
{
    n.ped <- nrow(sib1$pat)
    n.mark <- ncol(sib1$pat)
    n.al <- length(f)
    par.al <- list()
    for(par in 1:4) par.al[[par]] <-
        matrix(sample(1:n.al,n.ped*n.mark,
            replace=TRUE,prob=f),n.ped,n.mark)
    a <- inhe <- c(sib1,sib2)
    for (v in 1:4) for (par in 1:4)
        a[[v]][inhe[[v]]==par] <-
            par.al[[par]][inhe[[v]]==par]
    geno <- genotype(a[[1]],a[[2]],a[[3]],a[[4]],n.al)
    return(geno)
}

geno.given.ibd <- function(f=c(0.5,0.5))
{
    n.al <- length(f)
    P.0 <- outer(outer(f,f),outer(f,f))
    P.1 <- P.2 <- array(0,dim=rep(n.al,4))
    for(a2 in 1:n.al) for(a1 in 1:n.al)
    for(a3 in 1:n.al) for(a4 in 1:n.al)
    {
        if (a1==a3 & a2==a4)
        {
            P.2[a1,a2,a3,a4] <- f[a1]*f[a2]
            P.1[a1,a2,a3,a4] <- f[a1]*f[a2]*(f[a1]+f[a2])/2
        }
        if (a1==a3 & a2!=a4)
        {
            P.1[a1,a2,a3,a4] <- f[a1]*f[a2]*f[a4]/2
        }
        if (a1!=a3 & a2==a4)
        {
            P.1[a1,a2,a3,a4] <- f[a1]*f[a3]*f[a2]/2
        }
    }
    a1 <- rep(1:n.al,n.al^3)
    a2 <- rep(rep(1:n.al,rep(n.al,n.al)),n.al^2)
    a3 <- rep(rep(1:n.al,rep(n.al^2,n.al)),n.al)
    a4 <- rep(1:n.al,rep(n.al^3,n.al))
    geno <- genotype(a1,a2,a3,a4,n.al)
    P.0 <- tapply(as.vector(P.0),geno,sum)
    P.1 <- tapply(as.vector(P.1),geno,sum)
    P.2 <- tapply(as.vector(P.2),geno,sum)
    P <- cbind(P.0,P.1,P.2)
    colnames(P) <- paste("State=",0:2,sep="")
    return(P)
}

# Question 3
trans.mat <- function(theta)
{
    phi <- 1-theta^2-(1-theta)^2
    Tr <- matrix(c((1-phi)^2, 2*phi*(1-phi), phi^2,
           phi*(1-phi), phi^2 + (1-phi)^2, phi*(1-phi),
           phi^2, 2*phi*(1-phi), (1-phi)^2),3,3,byrow=TRUE)
    colnames(Tr) <- paste("to.IBD=",0:2,sep="")
    rownames(Tr) <- paste("from.IBD=",0:2,sep="")
    return(Tr)
}

theta <- 0.5 - 0.5*exp(-0.02*20)
round(trans.mat(theta),3)

# Question 4
Pr <- c(0.25,0.5,0.25)
Pr%*%trans.mat(theta)

# Question 5
forward <- function(G.I,Tr,Pr)
{
    n.samp <- dim(G.I)[1]
    n.mark <- dim(G.I)[2]
    F <- G.I
    F[,1,] <- sweep(G.I[,1,],2,Pr,"*")
    for (i in 2:n.mark)
    {
        F[,i,] <- G.I[,i,]*(F[,i-1,]%*%Tr)
        S <- F[,i,1] + F[,i,2] + F[,i,3]
        F[,i,] <- sweep(F[,i,],1,S,"/")
    }
    return(F)
}

# Question 5
backward <- function(G.I,Tr,Pr)
{
    n.samp <- dim(G.I)[1]
    n.mark <- dim(G.I)[2]
    B <- G.I
    B[,n.mark,] <- 1
    for (i in seq(n.mark-1,1))
    {
        B[,i,] <- (G.I[,i+1,]*B[,i+1,])%*%t(Tr)
        S <- B[,i,1] + B[,i,2] + B[,i,3]
        B[,i,] <- sweep(B[,i,],1,S,"/")
    }
    return(B)
}

# Question 5
marginal.post <- function(F,B)
{
    P <- F*B
    S <- P[,,1]+P[,,2]+P[,,3]
    P <- sweep(P,1:2,S,"/")
    return(P)
}


n.rep <- 10^2
n.ped <- 10^3
Delta <- c(35,20,10,5,1)
ibd.est.null <- matrix(nrow=3,ncol=length(Delta))
colnames(ibd.est.null) <- paste("Delta=",Delta,sep="")
rownames(ibd.est.null) <- c("mean","var","mse")
cor.ibd <- vector(mode="list",length=length(Delta))
names(cor.ibd) <- paste("Delta=",Delta,sep="")
cor.est <- cor.ibd
# Question 7
P <- geno.given.ibd()
for(i in 1:length(Delta))
{
    markers <- seq(0,140,by=Delta[i])
    n.mark <- length(markers)
```

```
  locus <- ceiling(n.mark/2)
  theta <- 0.5 - 0.5*exp(-0.02*Delta[i])
  Tr <- trans.mat(theta)
# Question 1
  fa <- list(pat=matrix(1,n.ped,n.mark),
             mat=matrix(2,n.ped,n.mark))
  mo <- list(pat=matrix(3,n.ped,n.mark),
             mat=matrix(4,n.ped,n.mark))
  ibd.est <- ibd <- NULL
# Question 6
  G.I <- array(dim=c(n.ped,n.mark,3))
  for (rep in 1:n.rep)
  {
# Question 1
    sib1 <- mating(fa,mo,markers)
    sib2 <- mating(fa,mo,markers)
    geno <- ped.geno(sib1,sib2)
# Question 6
    for(k in 1:3) G.I[,,k] <- matrix(P[geno,k],n.ped,n.mark)
    F.P <- forward(G.I,Tr,Pr)
    B.P <- backward(G.I,Tr,Pr)
    I.G <- marginal.post(F.P,B.P)
 # Question 7
   ibd.est <- rbind(ibd.est,2*I.G[,,3]+I.G[,,2])
    ibd <- rbind(ibd,(sib1$pat == sib2$pat)+
            (sib1$mat == sib2$mat))
  }
  ibd.est.null["mean",i] <- mean(ibd.est[,locus])
  ibd.est.null["var",i] <- var(ibd.est[,locus])
  ibd.est.null["mse",i] <- mean((ibd[,locus]-ibd.est[,locus])^2)
  cor.ibd[[i]] <- cor(ibd)
  cor.est[[i]] <- cor(ibd.est)
}
round(ibd.est.null,3)
```