University of Edinburgh
School of Mathematics
Incomplete Data Analysis

Supporting Materials for Lecture 8

ABO blood type example

The code for this example is pretty much similar, with the specific changes, to the one used in the genetic linkage example (lecture 7).

```
rm(list=ls())

blood_type_em=function(y,p0,eps){
p=p0
pa=p[1]; pb=p[2]
na=y[1]; nb=y[2]; nab=y[3]; no=y[4]; n=sum(y)
diff=1
while(diff>eps){
p.old=p
#E-step
naat=(na*pa)/(2-2*pb-pa)
nbbt=(nb*pb)/(2-2*pa-pb)
#M-step
pa=(naat+na+nab)/(2*n)
pb=(nbbt+nb+nab)/(2*n)
p=c(pa,pb)
diff=sum(abs(p-p.old))
}
return(p)
}

yobs=c(186,38,13,284)

res=blood_type_em(yobs,c(0.001,0.95),0.000001)
res

## [1] 0.21359090 0.05014533
```

We thus have $p_A = 0.214$, $p_B = 0.050$ and $p_O = 1 - 0.214 - 0.050 = 0.736$.

The bootstrap

Genetic linkage example

We start revisiting the code implementing the EM algorithm for this example.

```
rm(list=ls())

#em code for genetic linkage example
multi=function(y,theta0,eps){
n=sum(y); diff=1
theta=theta0
while(diff>eps){
theta.old=theta
#E step
zt=y[1]*0.5/(0.5+0.25*theta)
#M step
theta=(y[1]+y[4]-zt)/(n-zt)
diff=abs(theta-theta.old)
}
return(theta)
}

#original data
y=c(125,18,20,34); n=sum(y)

#estimate based on the original data
thetamle=multi(y=y,0.1,0.00001)
thetamle

## [1] 0.6268208
```

We will start implementing a parametric bootstrap. To this end, we will generate data from a multinomial distribution with probabilities $0.5+\widehat{\theta}_{\text{MLE}}/4$, $(1-\widehat{\theta}_{\text{MLE}})/4$, $(1-\widehat{\theta}_{\text{MLE}})/4$, and $\widehat{\theta}_{\text{MLE}}/4$. To simulate multinomially distributed data we will use the package `Hmisc`.

```
set.seed(1)
require(Hmisc)

B=2000 #number of bootstrap replicates
p=c(0.5+thetamle/4,(1-thetamle)/4,(1-thetamle)/4,thetamle/4)

ybp=matrix(0,nrow=n,ncol=B) #matrix to store the bootstrap samples
for(l in 1:B){
ybp[,l]=rMultinom(probs=rbind(p),n)
}
```

Having generated the data, we will now apply the above created function to each of the bootstrap generated datasets (thus having at the end B=2000 estimates). Since the function takes as imput the number of observations falling in each of the four categories, we will use the funcion `table`. As an example, for the first bootstrap dataset

```
ybp[,1]

##   [1] 1 1 1 4 1 4 4 2 1 1 1 1 2 1 3 1 2 4 1 3 4 1 1 1 1 1 1 1 4 1 1 1 1 1 3
##  [36] 2 3 1 2 1 3 1 3 1 1 3 1 1 2 2 1 4 1 1 1 1 1 1 2 1 4 1 1 1 1 1 1 3 1 4
##  [71] 1 3 1 1 1 4 4 1 3 4 1 2 1 1 3 1 2 1 1 1 1 1 1 4 3 3 1 1 3 1 1 1 1 4 1
```

```
## [106] 1 1 1 4 1 4 2 1 1 1 1 2 1 1 1 4 1 1 1 3 1 1 1 1 1 1 1 1 4 1 1 1 4 1
## [141] 2 1 1 1 2 1 1 2 1 4 1 1 1 1 1 1 1 1 1 1 4 1 3 4 1 1 1 2 1 1 3 4 1 1
## [176] 4 1 2 1 4 1 1 4 1 4 1 3 2 4 1 2 1 1 4 1 1 1

table(ybp[,1])

##
##   1   2   3   4
## 131  19  18  29

thetastarp=numeric(B)
for(i in 1:B){
thetastarp[i]=multi(as.numeric(table(ybp[,i])),0.1,0.00001)
}

sd(thetastarp); quantile(thetastarp,c(0.025,0.975))

## [1] 0.05280962
##      2.5%     97.5%
## 0.5184935 0.7288739
```

We will now apply the nonparametric bootstrap. In order to do that, we re-write the original data in an alternative way. We then use the empirical cdf (via the command `sample`) to bootstrap the data in a nonparametric way.

```
ystar=c(rep(1,125),rep(2,18),rep(3,20),rep(4,34))

yb=matrix(0,nrow=n,ncol=B)
for(l in 1:B){
yb[,l]=sample(ystar,size=n,replace=T)
}

thetastar=numeric(B)
for(i in 1:B){
thetastar[i]=multi(as.numeric(table(yb[,i])),0.1,0.00001)
}

sd(thetastar); quantile(thetastar,c(0.025,0.975))

## [1] 0.0512441
##      2.5%     97.5%
## 0.5229345 0.7198168
```

Old faithful dataset - mixture of normals example

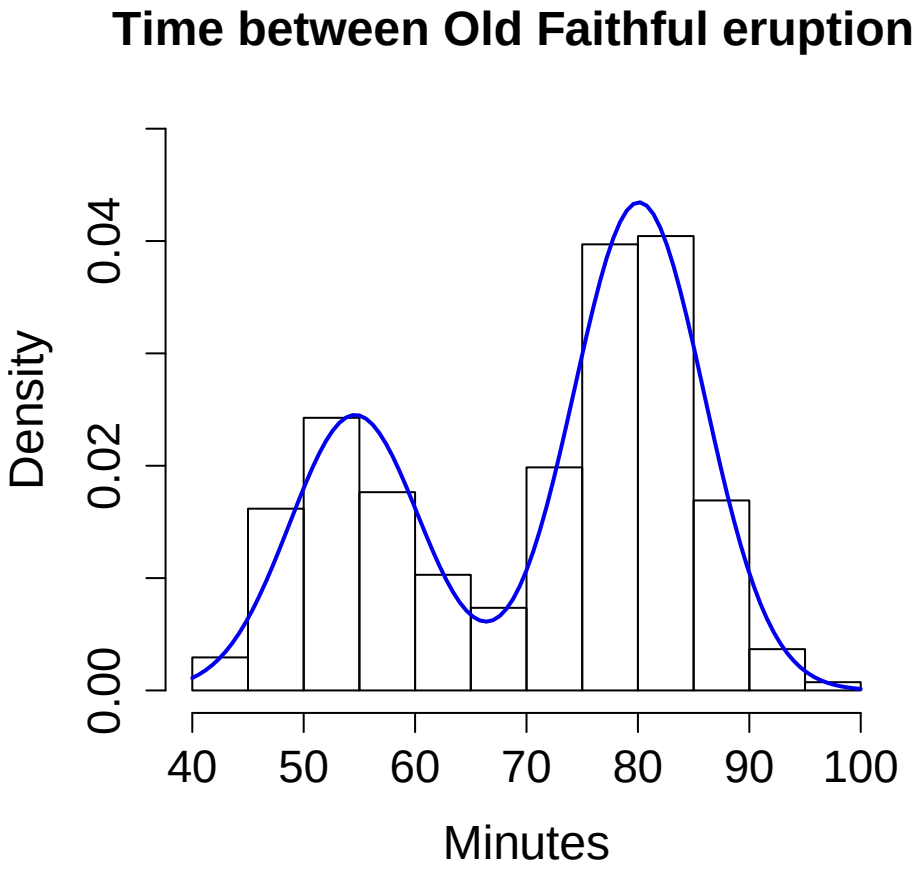
As in the previous case, we start by revisiting the code that implements the EM algorithm for a mixture of two normals.

```r
em.mixture.two.normal=function(y,theta0,eps){
n=length(y)
theta=theta0
p=theta[1]; mu1=theta[2]; sigma1=theta[3]; mu2=theta[4]; sigma2=theta[5]
diff=1
while(diff>eps){
theta.old=theta
#E-step
ptilde1=p*dnorm(y,mean=mu1,sd=sigma1)
ptilde2=(1-p)*dnorm(y,mean=mu2,sd=sigma2)
ptilde=ptilde1/(ptilde1+ptilde2)
#M-step
p=mean(ptilde)
mu1=sum(y*ptilde)/sum(ptilde)
sigma1=sqrt(sum(((y-mu1)^2)*ptilde)/sum(ptilde))
mu2=sum(y*(1-ptilde))/sum(1-ptilde)
sigma2=sqrt(sum(((y-mu2)^2)*(1-ptilde))/sum(1-ptilde))
theta=c(p,mu1,sigma1,mu2,sigma2)
diff=sum(abs(theta-theta.old))
}
return(theta)
}

data(faithful)
attach(faithful)

res=em.mixture.two.normal(y=waiting,c(0.4,40,4,90,4),0.00001)
p=res[1]; mu1=res[2]; sigma1=res[3]; mu2=res[4]; sigma2=res[5]

hist(waiting, main="Time between Old Faithful eruptions",xlab="Minutes", ylab="Density", cex.main=1.5, cex.la
curve(p*dnorm(x,mu1,sigma1)+(1-p)*dnorm(x,mu2,sigma2),add=TRUE,lwd=2,col="blue2")
```



We wil now implement a nonparametric bootstrap in a similar fashion to what we have done in the previous example.

```
B=1000; y=waiting; n=length(y)
yb=matrix(0,nrow=n,ncol=B); estimates=matrix(0,nrow=5,ncol=B)
for(l in 1:B){
yb[,l]=sample(y,size=n,replace=T)
estimates[,l]=em.mixture.two.normal(yb[,l],c(0.4,40,4,90,4),0.00001)
}

ql=function(x){quantile(x,0.025)}; qh=function(x){quantile(x,0.975)}
pl=ql(estimates[1,]); ph=qh(estimates[1,])
mu1l=ql(estimates[2,]); mu1h=qh(estimates[2,])
mu2l=ql(estimates[4,]); mu2h=qh(estimates[4,])
sigma1l=ql(estimates[3,]); sigma1h=qh(estimates[3,])
sigma2l=ql(estimates[5,]); sigma2h=qh(estimates[5,])
pl; ph

##      2.5%
## 0.2979489
##     97.5%
## 0.4206767
```

```
mu1l; mu1h

##     2.5%
## 53.07557
##    97.5%
## 56.26323

mu2l; mu2h

##     2.5%
## 78.95012
##    97.5%
## 81.05348

sigma1l; sigma1h

##    2.5%
## 4.89655
##    97.5%
## 6.788366

sigma2l; sigma2h

##     2.5%
## 5.075431
##    97.5%
## 6.699694
```