

Computational Statistics. Chapter 3: EM algorithm. Solution of exercises

Thierry Denoeux

9/6/2021

Exercise 1

Question 1a

We first give values to the model parameters:

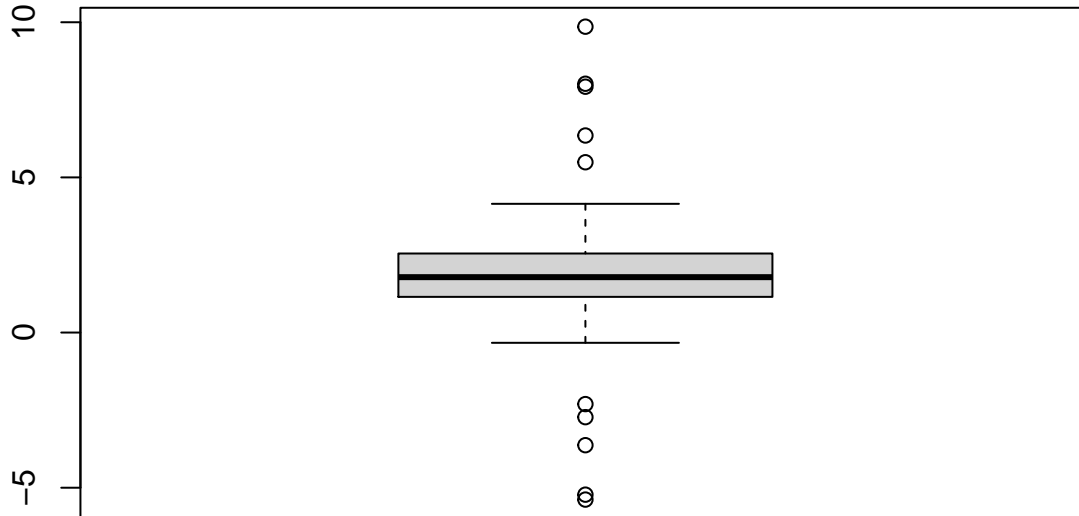
```
pi<-0.90
mu<-2
sig<- 1
a=10;
c<-1/(2*a)
n<- 100
```

We then generate the data:

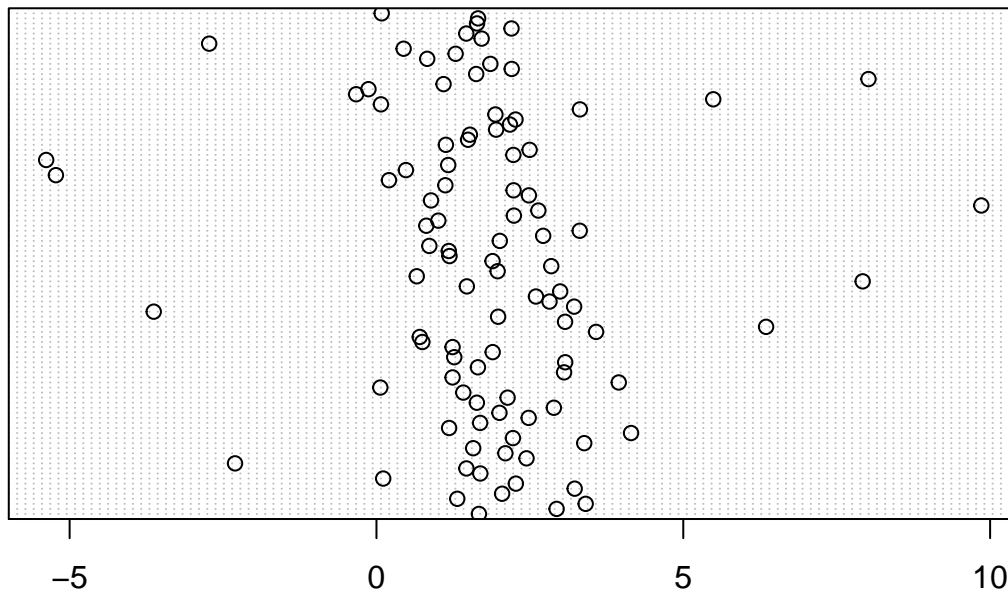
```
y<-vector("numeric",n)
z<-vector("numeric",n)
for(i in 1:n){
  z[i]=sample(c(1,0),size=1,prob=c(pi,1-pi))
  if(z[i]==1){
    y[i]<- rnorm(1,mean=mu,sd=sig)
  }
  else y[i]<-runif(1,min=-a,max=a)
}
```

Finally, we generate box and dot plots the data:

```
boxplot(y)
```



```
dotchart(y)
```



Question 1b

We first write a function that computes the observed-data log-likelihood:

```
loglik<- function(theta,y){
  phi <- sapply(y,dnorm,mean=theta[1],sd=theta[2])
  logL <- sum(log(theta[3]*phi+(1-theta[3])*c))
  return(logL)
}
```

We then write the EM algorithm for this problem. The inputs are the data y , the initial parameter value θ_0 , the constant a , the threshold ϵ used in the stopping criterion, and a flag `disp` that controls the display of the intermediate results. The outputs are the maximum observed-data log-likelihood, the corresponding MLE of θ , and the vector z of estimated probabilities.

```

em_outlier <- function(y,theta0,a,epsi,disp=TRUE){
  go_on<-TRUE
  logL0<- loglik(theta0,y)
  t<-0
  c<-1/(2*a)
  n<-length(y)
  if(disp) print(c(t,logL0))
  while(go_on){
    t<-t+1
    # E-step
    phi <- sapply(y,dnorm,mean=theta0[1],sd=theta0[2])
    z<- phi*theta0[3]/(phi*theta0[3]+c*(1-theta0[3]))
    # M-step
    S<- sum(z)
    pi<-S/n
    mu<- sum(y*z)/S
    sig<-sqrt(sum(z*(y-mu)^2)/S)
    theta<-c(mu,sig,pi)
    logL<-loglik(theta,y)
    if (logL-logL0 < epsi) go_on <- FALSE
    logL0 <- logL
    theta0<-theta
    if(disp) print(c(t,logL))
  }
  return(list(loglik=logL,theta=theta,z=z))
}

```

Question 1c

Let us now run the above function with our data. We initialize parameters μ and σ with the mean and standard deviation of the data, and we set $\pi_0 = 0.5$:

```

mu0<-mean(y) # +rnorm(1,mean=0,sd=0.5)
sig0<-sd(y)
pi0<-0.5
theta0<-c(mu0,sig0,pi0)

```

We then run function `em_outlier`:

```

estim<-em_outlier(y,theta0,a,epsi=1e-6)

```

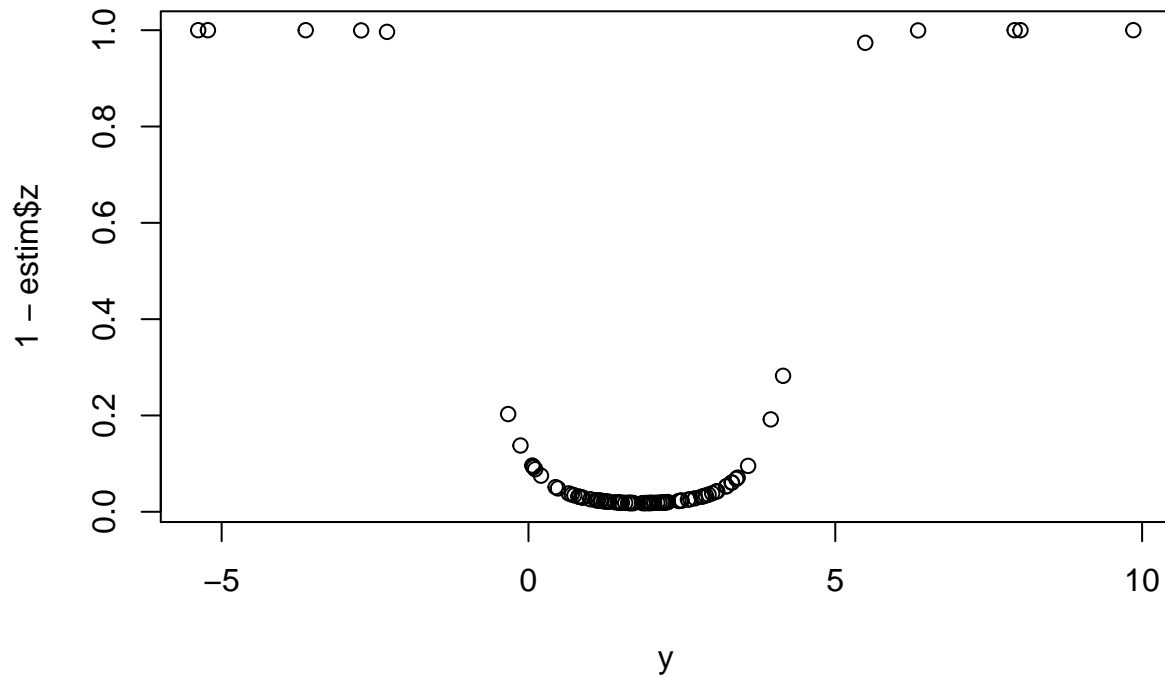
```

## [1] 0.0000 -232.4587
## [1] 1.0000 -191.3559
## [1] 2.0000 -184.0009
## [1] 3.0000 -183.6286
## [1] 4.0000 -183.5963
## [1] 5.0000 -183.5928
## [1] 6.0000 -183.5924
## [1] 7.0000 -183.5923
## [1] 8.0000 -183.5923
## [1] 9.0000 -183.5923

```

Finally, we plot the estimates probabilities $1 - z_i$ against the inputs y_i :

```
plot(y,1-estim$z)
```



We can see that the outliers have a high estimated probability of being drawn from the uniform distribution, as expected.

Exercise 2

Question 2a

We set the parameters:

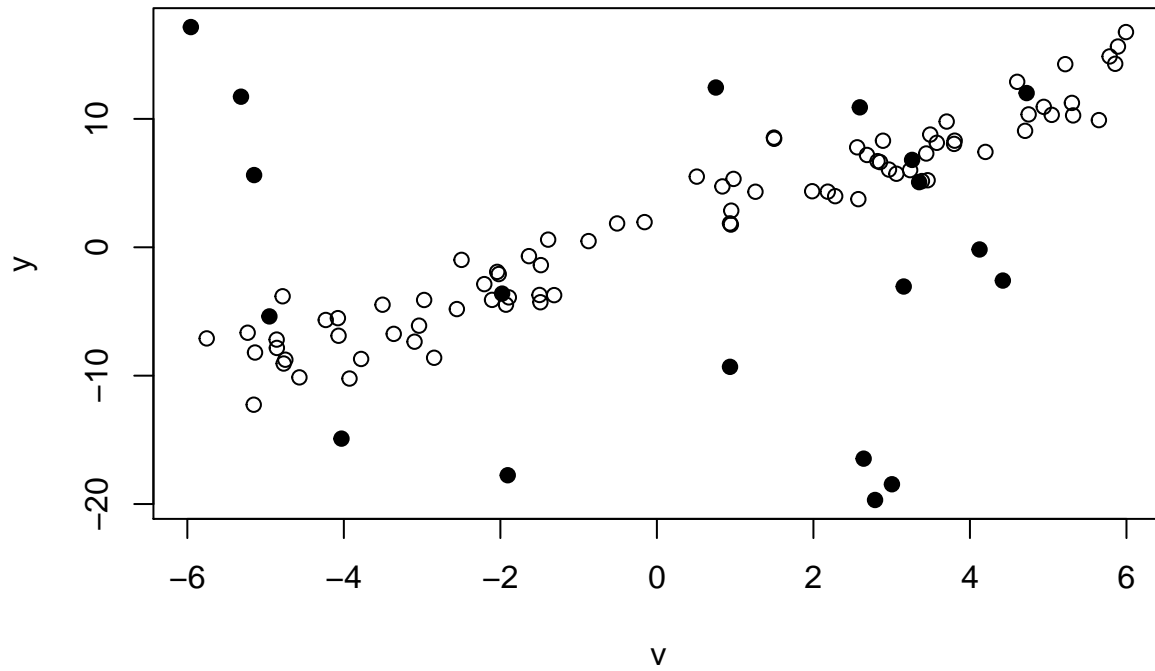
```
pi<-0.8
beta<-c(1,2)
sig<- 2
a=20
c<-1/(2*a)
n<- 100
```

We then generate the data

```
y<-vector("numeric",n)
v <- runif(n,min=-6,max=6)
z<-vector("numeric",n)
for(i in 1:n){
  z[i]=sample(c(1,0),size=1,prob=c(pi,1-pi))
  if(z[i]==1){
    y[i]<-rnorm(1,mean=beta[1]+v[i]*beta[2],sd=sig)
  } else y[i]<-runif(1,min=-a,max=a) }
```

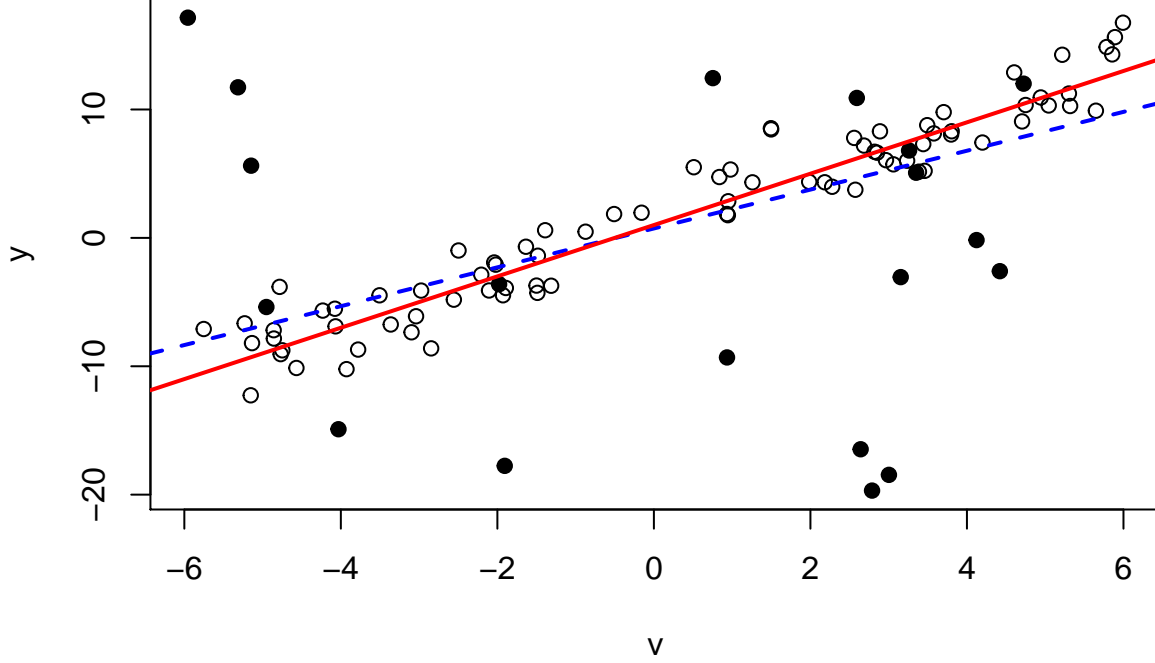
Finally, we plot the data. The data points generated from the uniform distribution (outliers) are highlighted:

```
plot(v,y)
points(v[z==0],y[z==0],pch=16)
```



Question 2b

```
reg<-lm(y~v)
plot(v,y)
points(v[z==0],y[z==0],pch=16)
abline(reg,lty=2,col="blue",lwd=2) # LS line
abline(1,2,lty=1,col="red",lwd=2) #true regression line
```



Question 2c

In this problem the observed data is the vector $\mathbf{y} = (y_1, \dots, y_n)$. The observed-data likelihood is

$$L(\boldsymbol{\theta}) = \prod_{i=1}^n f(y_i; \boldsymbol{\theta}) = \prod_{i=1}^n [\pi \phi(y_i, v_i^T \boldsymbol{\beta}, \sigma) + (1 - \pi)c].$$

The missing data is the random vector $\mathbf{Z} = (Z_1, \dots, Z_n)$, where $Z_i = 1$ if the observation is an outlier, and $Z_i = 0$ otherwise. Clearly, Z_i has a Bernoulli distribution $\mathcal{B}(1 - \pi)$. The complete-data likelihood is

$$L_c(\boldsymbol{\theta}) = \prod_{i=1}^n f(y_i | z_i; \boldsymbol{\theta}) f(z_i; \boldsymbol{\theta}) = \prod_{i=1}^n [\phi(y_i, v_i^T \boldsymbol{\beta}, \sigma)^{z_i} c^{1-z_i} \pi^{z_i} (1 - \pi)^{1-z_i}],$$

and its logarithm is

$$\ell_c(\boldsymbol{\theta}) = \sum_{i=1}^n [z_i \log \phi(y_i, v_i^T \boldsymbol{\beta}, \sigma) + (1 - z_i) \log c + z_i \log \pi + (1 - z_i) \log(1 - \pi)].$$

Function Q is obtained by replacing the terms z_i by their conditional expectations given $\mathbf{Y} = \mathbf{y}$, for $\boldsymbol{\theta} = \boldsymbol{\theta}^{(t)}$. In the E-step, we will compute these conditional expectations as

$$z_i^{(t)} = \mathbb{E}_{\boldsymbol{\theta}^{(t)}}[Z_i | y_i] = \mathbb{P}_{\boldsymbol{\theta}^{(t)}}[Z_i = 1 | y_i] = \frac{\phi(y_i; v_i^T \boldsymbol{\beta}^{(t)}, \sigma^{(t)}) \pi^{(t)}}{\phi(y_i; v_i^T \boldsymbol{\beta}^{(t)}, \sigma^{(t)}) \pi^{(t)} + c(1 - \pi^{(t)})} \quad (1)$$

and

$$\begin{aligned} Q(\boldsymbol{\theta}, \boldsymbol{\theta}^{(t)}) &= \sum_{i=1}^n \left[z_i^{(t)} \log \phi(y_i, v_i^T \boldsymbol{\beta}, \sigma) + (1 - z_i^{(t)}) \log c + z_i^{(t)} \log \pi + (1 - z_i^{(t)}) \log(1 - \pi) \right] \\ &= \underbrace{\sum_{i=1}^n z_i^{(t)} \log \phi(y_i, v_i^T \boldsymbol{\beta}, \sigma)}_A + \underbrace{\log \pi \sum_{i=1}^n z_i^{(t)} + \log(1 - \pi) \left(n - \sum_{i=1}^n z_i^{(t)} \right)}_B + \underbrace{(1 - z_i^{(t)}) \log c}_C. \end{aligned}$$

In the M-step, we maximize $Q(\boldsymbol{\theta}, \boldsymbol{\theta}^{(t)})$ with respect to $\boldsymbol{\theta}$. In the above equation A depends only on β and σ , B depends on π , and C is a constant. We can thus maximize A and B separately. We have

$$A = -\frac{1}{2} \left(\sum_{i=1}^n z_i^{(t)} \right) \log(2\pi) - \left(\sum_{i=1}^n z_i^{(t)} \right) \log \sigma - \frac{1}{2\sigma^2} \sum_{i=1}^n z_i^{(t)} (y_i - v_i^T \beta)^2.$$

We can see that the vector β maximizing A (and Q) is the solution of a weighted least-squares problem (each individual error term being weighted by $z_i^{(t)}$). We thus have the following update rule for β :

$$\beta^{(t+1)} = \left[\sum_{i=1}^n z_i^{(t)} v_i v_i^T \right]^{-1} \sum_{i=1}^n z_i^{(t)} v_i y_i.$$

To maximize A w.r.t. σ , we set $\beta = \beta^{(t+1)}$ and we compute the derivative:

$$\frac{\partial A}{\partial \sigma} = -\frac{1}{\sigma} \sum_i z_i^{(t)} + \frac{1}{\sigma^3} \sum_i z_i^{(t)} (y_i - v_i^T \beta^{(t+1)})^2.$$

Setting this derivative to zero, we get the update equation for σ :

$$\sigma^{(t+1)} = \sqrt{\frac{\sum_{i=1}^n z_i^{(t)} (y_i - v_i^T \beta^{(t+1)})^2}{\sum_{i=1}^n z_i^{(t)}}}.$$

Finally, to maximize B , we compute the compute the derivative

$$\frac{\partial B}{\partial \pi} = \frac{\sum_{i=1}^n z_i^{(t)}}{\pi} - \frac{n - \sum_{i=1}^n z_i^{(t)}}{1 - \pi}.$$

Setting this derivative to zero we get the update rule for π :

$$\pi^{(t+1)} = \frac{1}{n} \sum_{i=1}^n z_i^{(t)}.$$

We can now write an EM algorithm for this problem. We first write a function that computes the observed-data log-likelihood:

```
loglik_reg<- function(theta,y,v){
  n<-length(y)
  phi<-vector(n,mode="numeric")
  for(i in 1:n) phi[i] <- dnorm(y[i],mean=theta[1]+theta[2]*v[i],sd=theta[3])
  logL <- sum(log(theta[4]*phi+(1-theta[4])*c))
  return(logL)
}
```

We then write the EM algorithm, which is similar to that of Exercise 1:

```
em_outlier_reg <- function(y,v,theta0,a,epsi){
  go_on<-TRUE
  logL0<- loglik_reg(theta0,y,v)
  t<-0
  c<-1/(2*a)
  n<-length(y)
  phi<-vector(n,mode="numeric")
  print(c(t,logL0))
  while(go_on){
```

```

t<-t+1
# E-step
for(i in 1:n) phi[i] <- dnorm(y[i],mean=theta0[1]+theta0[2]*v[i],sd=theta0[3])
z<- phi*theta0[4]/(phi*theta0[4]+c*(1-theta0[4]))
# M-step
S<- sum(z)
pi<-S/n
reg<-lm(y ~v,weights=z)
beta<-reg$coefficients
sig<-sqrt(sum(z*reg$residuals^2)/S)
theta<-c(beta,sig,pi)
logL<-loglik_reg(theta,y,v)
if (logL-logL0 < epsi) go_on <- FALSE
logL0 <- logL
theta0<-theta
print(c(t,logL))
}
return(list(loglik=logL,theta=theta,z=z))
}

```

Question 2d

We first initialize the parameters to the OLS estimates and some arbitrary value of π :

```

beta0<-reg$coefficients
sig0<-sd(reg$residuals)
pi0<-0.8
theta0<-c(beta0,sig0,pi0)

```

We then run function `em_outlier_reg`:

```

estim<-em_outlier_reg(y,v,theta0,a,epsi=1e-6)

```

```

## [1] 0.0000 -319.6239
## [1] 1.0000 -286.311
## [1] 2.0000 -270.7526
## [1] 3.0000 -266.9142
## [1] 4.0000 -266.793
## [1] 5.0000 -266.7853
## [1] 6.0000 -266.7845
## [1] 7.0000 -266.7844
## [1] 8.0000 -266.7844
## [1] 9.0000 -266.7844
## [1] 10.0000 -266.7844

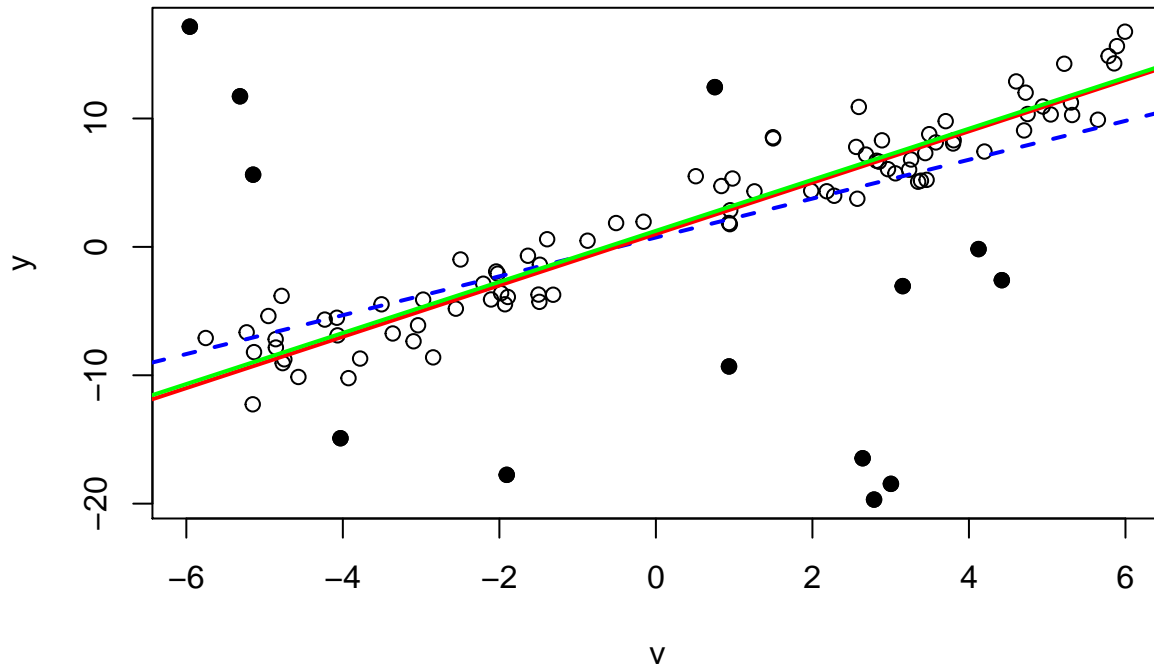
```

Finally, we plot the the line estimated by EM, together with the true regression line and the OLS line. We also plot the points identified as outliers:

```

plot(v,y)
abline(reg,lty=2,col="blue",lwd=2) # LS line
abline(1,2,lty=1,col="red",lwd=2) # regression line
abline(estim$theta[1],estim$theta[2],lty=1,col='green',lwd=2) # line estimated using EM
points(v[estim$z<0.5],y[estim$z<0.5],pch=19) # points identified as outliers

```

Exercise 3

Question 3a

We write a function 'gen_data\$ that generates a sample:

```
gen_sample<- function(mu,sig,pi,a,n){
  y<-rep(0,n)
  z<-rep(0,n)
  for(i in 1:n){
    z[i]=sample(c(1,0),size=1,prob=c(pi,1-pi))
    if(z[i]==1) y[i]<- rnorm(1,mean=mu,sd=sig) else y[i]<-runif(1,min=-a,max=a)
  }
  return(y)
}
```

We then define true values of the parameters:

```
pi<-0.90
mu<-0
sig<- 1
theta_true<-c(mu,sig,pi)
```

and generate $N = 1000$ and estimate θ from each sample using function `em_outlier`; the estimates $\hat{\theta}$ are stored in a matrix `Theta`:

```
a=5
c<-1/(2*a)
n<- 100
N<-1000
Theta<-matrix(0,N,3)
pi0<-0.5
```

```

for(j in 1:N){
  y<- gen_sample(mu,sig,pi,a,n)
  mu0<-mean(y)
  sig0<-sd(y)
  theta0<-c(mu0,sig0,pi0)
  em<-em_outlier(y,theta0,a,epsi=1e-5,disp=FALSE)
  Theta[j,]<-em$theta }

```

Finally, we can estimate the covariance matrix of $\hat{\theta}$:

```

V<-var(Theta)
print(V,2)

```

```

##          [,1]      [,2]      [,3]
## [1,] 0.01292 0.00032 0.00033
## [2,] 0.00032 0.01130 0.00252
## [3,] 0.00033 0.00252 0.00270

```

Question 3b

We first need to compute the complete-data information matrix

$$\hat{i}_{\mathbf{x}}(\hat{\theta}) = \mathbb{E}_{\hat{\theta}}(-\ell''(\hat{\theta}) \mid \curvearrowright).$$

The complete-data likelihood is

$$\begin{aligned} L_c(\theta) &= \prod_{i=1}^n f(y_i, z_i; \theta) = \prod_{i=1}^n f(y_i \mid z_i; \mu, \sigma) f(z_i; \pi) \\ &= \prod_{i=1}^n [\phi(y_i; \mu, \sigma)^{z_i} c^{1-z_i} \pi^{z_i} (1-\pi)^{1-z_i}] \end{aligned}$$

and the corresponding log-likelihood is:

$$\ell_c(\theta) = \sum_{i=1}^n z_i \log \phi(y_i; \mu, \sigma) + \left(n - \sum_{i=1}^n z_i \right) \log c + \sum_{i=1}^n (z_i \log \pi + (1 - z_i) \log(1 - \pi))$$

We need to compute the first and then the second derivative. We have

$$\begin{aligned} \frac{\partial \ell_c}{\partial \mu} &= \frac{1}{\sigma^2} \sum_i z_i (y_i - \mu) \\ \frac{\partial \ell_c}{\partial \sigma} &= -\frac{1}{\sigma} \sum_i z_i + \frac{1}{\sigma^3} \sum_i z_i (y_i - \mu)^2 \\ \frac{\partial \ell_c}{\partial \pi} &= \sum_i \left(\frac{z_i}{\pi} - \frac{1 - z_i}{1 - \pi} \right) \end{aligned}$$

And then the second derivatives:

$$\begin{aligned} \frac{\partial^2 \ell_c}{\partial \mu^2} &= -\frac{1}{\sigma^2} \sum_i z_i \\ \frac{\partial^2 \ell_c}{\partial \mu \partial \sigma} &= -\frac{2}{\sigma^3} \sum_i z_i (y_i - \mu) \end{aligned}$$

$$\begin{aligned}\frac{\partial^2 \ell_c}{\partial \sigma^2} &= \frac{1}{\sigma^2} \sum_i z_i - \frac{3}{\sigma^4} \sum_i z_i (y_i - \mu)^2 \\ \frac{\partial^2 \ell_c}{\partial \pi^2} &= \sum_i \left(\frac{-z_i}{\pi^2} - \frac{1-z_i}{(1-\pi)^2} \right) = -\frac{\sum_i z_i}{\pi^2} - \frac{n - \sum_i z_i}{(1-\pi)^2} \\ \frac{\partial^2 \ell_c}{\partial \mu \partial \pi} &= \frac{\partial^2 \ell_c}{\partial \sigma \partial \pi} = 0.\end{aligned}$$

To compute the conditional expectation of $\ell_c(\boldsymbol{\theta}^{(t)})$ for $\boldsymbol{\theta} = \boldsymbol{\theta}^{(t)}$, we simply replace z_i by $z_i^{(t)}$ in the above expression. (When the EM algorithm has converged, $\boldsymbol{\theta}^{(t)} \approx \hat{\boldsymbol{\theta}}$). The complete information matrix is then

$$\hat{i}_{\mathbf{x}}(\hat{\boldsymbol{\theta}}) = \begin{pmatrix} \frac{1}{\sigma^2} \sum_i z_i^{(t)} & \frac{2}{\sigma^3} \sum_i z_i^{(t)} (y_i - \hat{\mu}) & 0 \\ \frac{2}{\sigma^3} \sum_i z_i^{(t)} (y_i - \hat{\mu}) & \frac{1}{\sigma^2} \sum_i z_i^{(t)} - \frac{3}{\sigma^4} \sum_i z_i^{(t)} (y_i - \hat{\mu})^2 & 0 \\ 0 & 0 & -\frac{\sum_i z_i^{(t)}}{\pi^2} - \frac{n - \sum_i z_i^{(t)}}{(1-\pi)^2} \end{pmatrix}.$$

The following R function computes the complete information matrix:

```
complete_information<- function(theta,y,z){
  I<-matrix(0,3,3)
  mu<-theta[1]
  sig<-theta[2]
  pi<-theta[3]
  S<-sum(z)
  I[1,1]<- S/sig^2
  I[1,2]<- 2/sig^3*sum(z*(y-mu))
  I[2,1]<-I[1,2]
  I[2,2]<- -S/sig^2 + 3/sig^4*sum(z*(y-mu)^2)
  I[3,3]<- S/pi^2 + (n-S)/(1-pi)^2
  return(I)
}
```

Next, we need to compute the missing information as the variance of the conditional score:

$$\hat{i}_{\mathbf{z}|\mathbf{Y}}(\boldsymbol{\theta}) = \text{Var} \left[\frac{\partial \log f(\mathbf{z} | \mathbf{y}; \boldsymbol{\theta})}{\partial \boldsymbol{\theta}} \mid \mathbf{y} \right].$$

For this, we first need to compute $f(\mathbf{z} | \mathbf{y}; \boldsymbol{\theta})$. We recall that

$$P(Z_i = 1 | y_i) = \frac{\phi_i \pi}{\phi(y_i; \mu, \sigma)_i \pi + c(1-\pi)} = p_i(\boldsymbol{\theta}).$$

with

$$\phi_i = \phi(y_i; \mu, \sigma) = \frac{1}{\sigma \sqrt{2\pi}} \exp \left[-\frac{1}{2\sigma^2} (y_i - \mu)^2 \right].$$

So,

$$f(\mathbf{z} | \mathbf{y}; \boldsymbol{\theta}) = \prod_{i=1}^n f(z_i | y_i; \boldsymbol{\theta}) = \prod_{i=1}^n p_i(\boldsymbol{\theta})^{z_i} (1 - p_i(\boldsymbol{\theta}))^{1-z_i},$$

and

$$\begin{aligned}\log f(\mathbf{z} | \mathbf{y}; \boldsymbol{\theta}) &= \sum_{i=1}^n (z_i (\log \phi_i + \log \pi - \log[\phi_i \pi + c(1-\pi)]) + (1-z_i) (\log c(1-\pi) - \log[\phi_i \pi + c(1-\pi)])) \\ &= \sum_{i=1}^n (z_i \log \phi_i + z_i \log \pi + (1-z_i) \log c(1-\pi) - \log[\phi_i \pi + c(1-\pi)]).\end{aligned}$$

To compute the score, we first compute:

$$\frac{\partial \phi_i}{\partial \mu} = \phi_i \frac{y_i - \mu}{\sigma^2},$$

and

$$\frac{\partial \phi_i}{\partial \sigma} = \frac{\phi_i}{\sigma} \left(\frac{(y_i - \mu)^2}{\sigma^2} - 1 \right).$$

and Using elementary calculus, we get

$$\begin{aligned} \frac{\partial f(\mathbf{z} | \mathbf{y}; \boldsymbol{\theta})}{\partial \mu} &= \frac{1}{\sigma^2} \sum_{i=1}^n z_i (y_i - \mu) - \frac{\pi}{\sigma^2} \sum_{i=1}^n \frac{(y_i - \mu) \phi_i}{\phi_i \pi + c(1 - \pi)} \\ \frac{\partial f(\mathbf{z} | \mathbf{y}; \boldsymbol{\theta})}{\partial \sigma} &= \frac{1}{\sigma} \sum_{i=1}^n z_i \left(\frac{(y_i - \mu)^2}{\sigma^2} - 1 \right) - \frac{\pi}{\sigma} \sum_{i=1}^n \frac{\phi_i \left(\frac{(y_i - \mu)^2}{\sigma^2} - 1 \right)}{\phi_i \pi + c(1 - \pi)} \\ \frac{\partial f(\mathbf{z} | \mathbf{y}; \boldsymbol{\theta})}{\partial \pi} &= \frac{\sum_{i=1}^n z_i - n\pi}{\pi(1 - \pi)} - \sum_{i=1}^n \frac{\phi_i - c}{\phi_i \pi + c(1 - \pi)}. \end{aligned}$$

The calculation of the score is performed by the following R function:

```
score<- function(theta,y,z,c){
  S<-c(0,0,0)
  mu<-theta[1]
  sig<-theta[2]
  pi<-theta[3]
  SZ<-sum(z)
  phi<- sapply(y,dnorm,mean=mu,sd=sig)
  S[1]<- 1/sig^2*sum(z*(y-mu)) - pi/sig^2*sum((y-mu)*phi/(phi*pi+c*(1-pi)))
  S[2]<- -SZ/sig + 1/sig^3*sum(z*(y-mu)^2) -
        pi/sig*sum( phi*((y-mu)^2/sig^2-1)/(phi*pi+c*(1-pi)) )
  S[3]<- (SZ-n*pi)/(pi*(1-pi)) - sum((phi-c)/(phi*pi+c*(1-pi)))
  return(S)
}
```

We can now compute the observed information for an observed sample. We first generate the data and compute the MLE:

```
set.seed(42)
y<- gen_sample(mu,sig,pi,a,n)
mu0<-mean(y)
sig0<-sd(y)
pi0<-0.5
theta0<-c(mu0,sig0,pi0)
em<-em_outlier(y,theta0,a,epsi=1e-5)
```

```
## [1] 0.0000 -191.3151
## [1] 1.0000 -173.8608
## [1] 2.0000 -169.5465
## [1] 3.0000 -168.7118
## [1] 4.0000 -168.4584
## [1] 5.0000 -168.3611
## [1] 6.0000 -168.3219
## [1] 7.0000 -168.3059
## [1] 8.0000 -168.2994
## [1] 9.0000 -168.2968
## [1] 10.0000 -168.2957
```

```
## [1] 11.0000 -168.2953
## [1] 12.0000 -168.2951
## [1] 13.0000 -168.295
## [1] 14.0000 -168.295
## [1] 15.0000 -168.295
## [1] 16.0000 -168.295
```

We then compute by simulation the missing information $\hat{i}_{Z|Y}(\theta)$ as the variance of the score:

```
N<-100000
SS<-matrix(0,N,3)
ix<-complete_information(em$theta,y,em$z)
for(i in (1:N)){
  u<-runif(n)
  Z<-as.numeric(u<=em$z)
  SS[i,]<-score(em$theta,y,Z,c)
}
izy<-var(SS)
```

Finally, we compute the observed information as the difference between the complete information and missing information; and we estimate the variance of $\hat{\theta}$ as the inverse of the observed information:

```
iy<-ix-izy
varh<-solve(iy)
print(varh,2)
```

```
##          [,1]  [,2]  [,3]
## [1,] 0.01254 -0.0011 -0.00047
## [2,] -0.00111 0.0086 0.00175
## [3,] -0.00047 0.0018 0.00315
```

To check our calculations, we can compare our result to that using numerical differentiation:

```
library(numDeriv)
varh1<-solve(-hessian(loglik,em$theta,y=y))
print(varh1,2)
```

```
##          [,1]  [,2]  [,3]
## [1,] 0.01254 -0.0011 -0.00046
## [2,] -0.00112 0.0087 0.00179
## [3,] -0.00046 0.0018 0.00318
```