

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

Thierry Denoeux

2024-02-20

Exercise 1

Question 1a

We first give values to the model parameters:

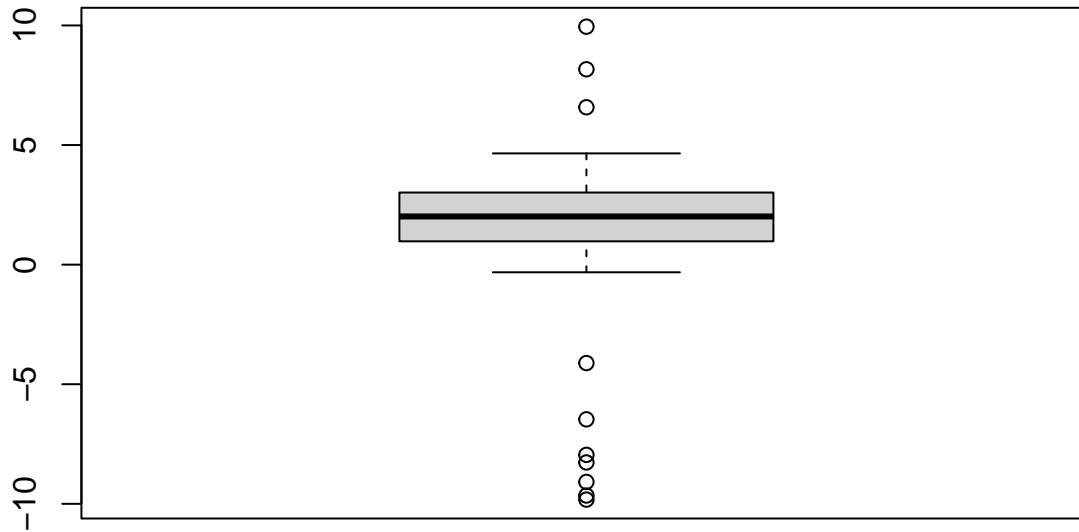
```
pi <- 0.90
mu <- 2
sig <- 1
a <- 10
c <- 1/(2*a)
n <- 100
```

We then generate the data:

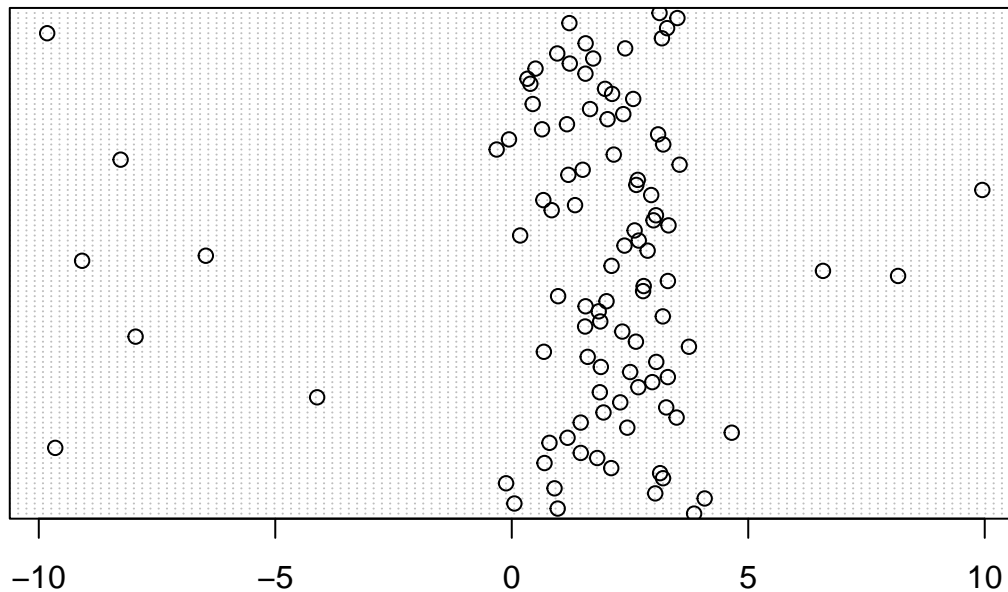
```
y<-vector("numeric",n)
z<-vector("numeric",n)
for(i in 1:n){
  z[i] <- sample(c(1,0),size=1,prob=c(pi,1-pi))
  if(z[i]==1)
    y[i] <- rnorm(1,mean=mu,sd=sig)
  else y[i] <- runif(1,min=-a,max=a)
}
```

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)

```

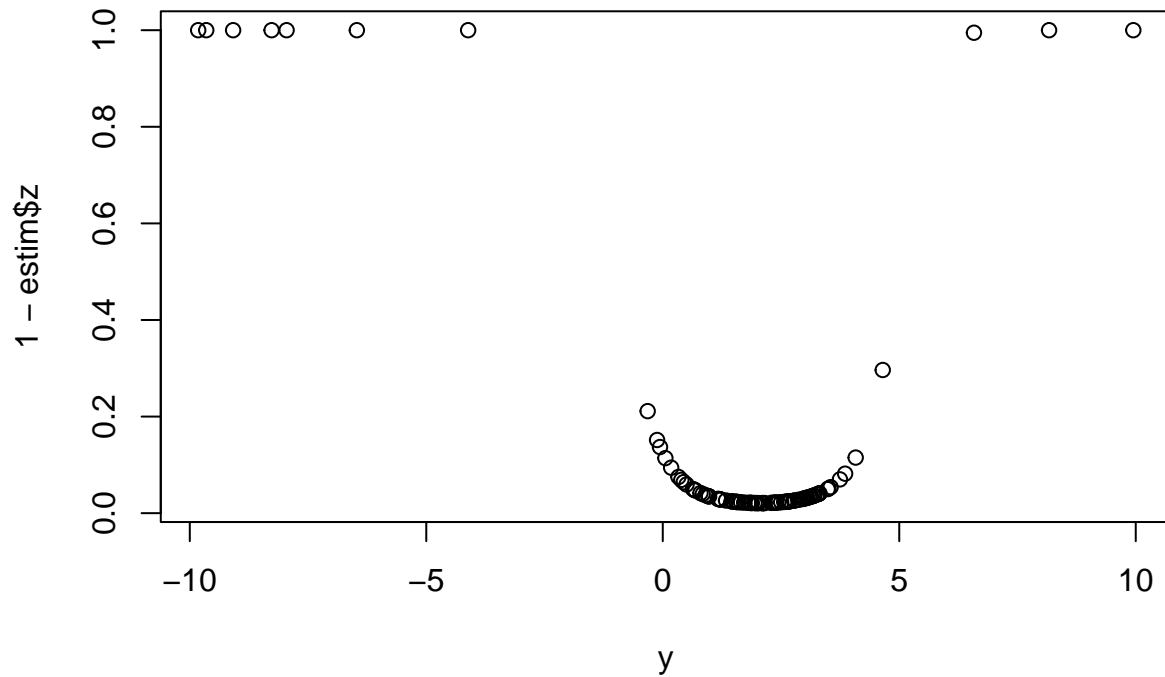
```

## [1] 0.0000 -256.7589
## [1] 1.0000 -206.1906
## [1] 2.0000 -195.1083
## [1] 3.0000 -194.1654
## [1] 4.0000 -194.0543
## [1] 5.0000 -194.0417
## [1] 6.0000 -194.0403
## [1] 7.0000 -194.0402
## [1] 8.0000 -194.0402
## [1] 9.0000 -194.0402
## [1] 10.0000 -194.0402

```

Finally, we plot the estimated 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:

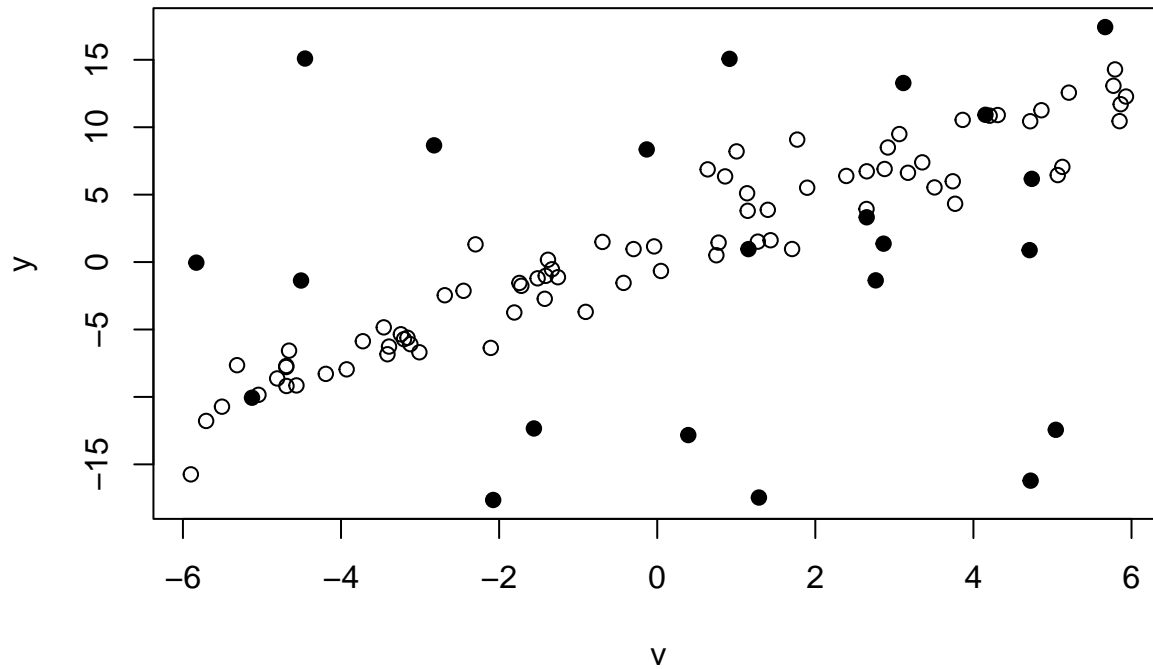
```
pi<-0.8
beta<-c(1,2)
sig<- 2
a=20
c<-1/(2*a)
n<- 100
```

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) }
```

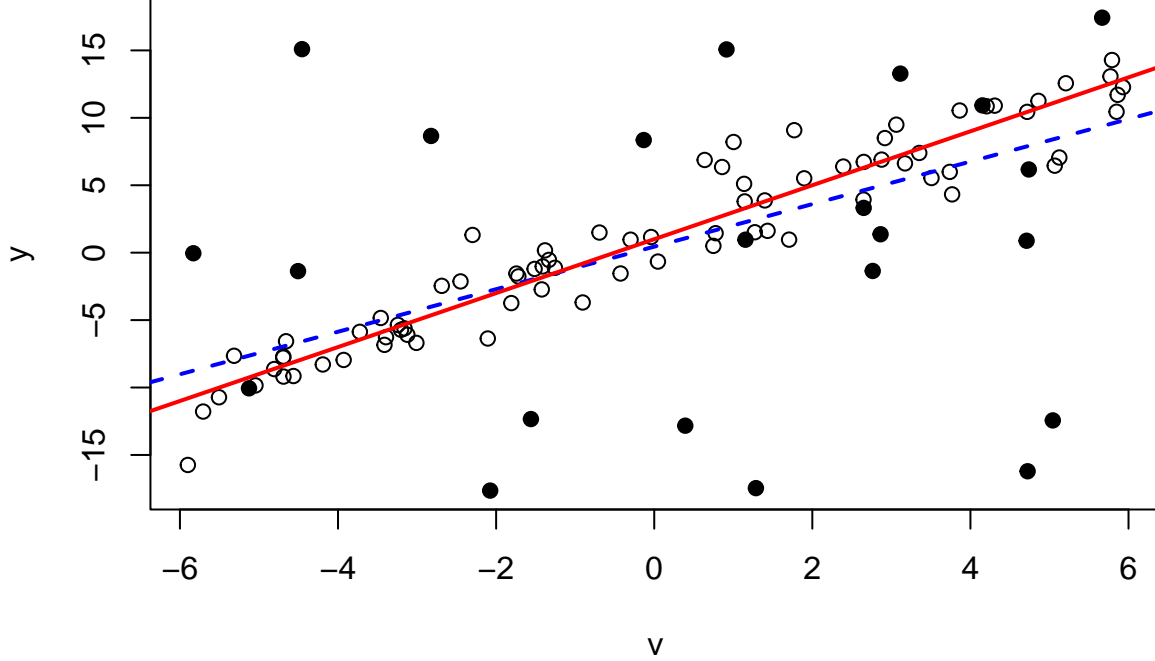
Finally, we plot the data. The data points generated from the uniform distribution (outliers) are highlighted:

```
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,eps){
  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,eps=1e-6)

```

```

## [1] 0.0000 -314.8432
## [1] 1.0000 -289.7889
## [1] 2.0000 -282.5777
## [1] 3.0000 -280.329
## [1] 4.0000 -279.5323
## [1] 5.0000 -279.2161
## [1] 6.0000 -279.0771
## [1] 7.0000 -279.0119
## [1] 8.0000 -278.9799
## [1] 9.0000 -278.9638
## [1] 10.0000 -278.9555
## [1] 11.0000 -278.9512
## [1] 12.0000 -278.9489
## [1] 13.0000 -278.9477
## [1] 14.0000 -278.9471
## [1] 15.0000 -278.9467
## [1] 16.0000 -278.9465
## [1] 17.0000 -278.9464
## [1] 18.0000 -278.9464
## [1] 19.0000 -278.9464
## [1] 20.0000 -278.9463

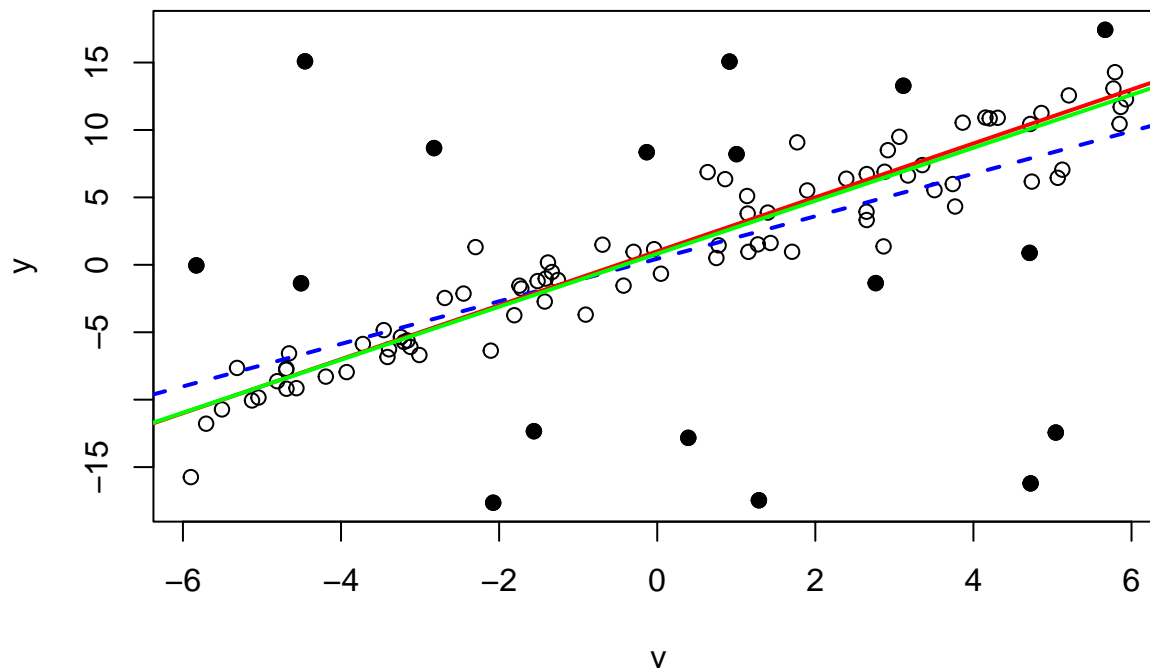
```



```
## [1] 21.0000 -278.9463
## [1] 22.0000 -278.9463
## [1] 23.0000 -278.9463
## [1] 24.0000 -278.9463
## [1] 25.0000 -278.9463
```

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)
  for(i in 1:n){
    z<-sample(c(1,0),size=1,prob=c(pi,1-pi))
    if(z==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: