刘程华 2018011687 计91

Prob 1. Derive the formulae on $\mathrm{p}.31$.

Gaussian Mixture Model EM

$\theta = \{\mu_1, \ldots, \mu_m, \sigma_1, \ldots, \sigma_m, \alpha_1, \ldots, \alpha_m\}$, and so from the first section of this note, our likelihood is:

$$L\left(\theta \mid X_1, \ldots, X_n\right) = \prod_{i=1}^{n} \sum_{k=1}^{m} \alpha_k N\left(x_i; \mu_k, \sigma_k^2\right)$$

So our log-likelihood is:

$$\ell(\theta) = \sum_{i=1}^{n} \log\left(\sum_{k=1}^{m} \alpha_k N\left(x_i; \mu_k, \sigma_k^2\right)\right) \tag{1}$$

Take initial guesses for the parameters $\widehat{\Theta}$

E step: compute the posterior distribution of z_i given the observations

$$\Lambda_{i,k} = P_{\hat{\theta}}\left(z_i = k \mid x_i\right) = \frac{\hat{\alpha}_k P_{\hat{\theta}_k}\left(x_i\right)}{P_{\hat{\theta}}\left(x_i\right)} = \frac{\hat{\alpha}_k P_{\hat{\theta}_k}\left(x_i\right)}{\sum_{j=1}^{m} \hat{\alpha}_j P_{\hat{\theta}_j}\left(x_i\right)} \tag{2}$$

$$Q(\Theta \mid \hat{\theta}) = \sum_{i=1}^{n} \sum_{k=1}^{m} \Lambda_{i,k} \log(\alpha_k P_{\theta_k}\left(x_i\right))$$

M step: update $\alpha_k, \mu_k, \sigma_k$ for each Gaussian $k = 1, \ldots, m$

$$\hat{\alpha}_k = \frac{1}{n} \sum_{i=1}^{n} \Lambda_{i,k}, \hat{\mu}_k = \frac{\sum_{i=1}^{n} x_i \Lambda_{i,k}}{\sum_{i=1}^{n} \Lambda_{i,k}}, \hat{\sigma}_k^2 = \frac{\sum_{i=1}^{n} \Lambda_{i,k} \left\| x_i - \hat{\mu}_k \right\|^2}{\sum_{i=1}^{n} \Lambda_{i,k}}$$

Proof

Let the partial derivative of (1) about μ_k equal to

$$\sum_{i=1}^{n} \left\{ \frac{1}{\sum_{k=1}^{m} \alpha_k N\left(x_i; \mu_k, \sigma_k\right)} \alpha_k N\left(x_i; \mu_k, \sigma_k\right) \frac{\left(x_i - \mu_k\right)}{\sigma_k^2} \right\} = 0 \tag{3}$$

by (2) and (3)

$$\sum_{i=1}^{n} \Lambda_{i,k} \frac{\left(x_i - \mu_k\right)}{\sigma_k^2} = 0$$

$$\Rightarrow \hat{\mu}_k = \frac{\sum_{i=1}^{n} x_i \Lambda_{i,k}}{\sum_{i=1}^{n} \Lambda_{i,k}}$$

Similarly, let the partial derivative of (1) about σ_k^2 , we have

$$\sum_{i=1}^{n}\left\{\frac{1}{\sum_{k=1}^{m}\alpha_k N\left(x_i;\mu_k,\sigma_k\right)}\alpha_k N\left(x_i;\mu_k,\sigma_k\right)\left(\frac{(x_i-\mu_k)^2}{2\sigma_k^4}-\frac{1}{2\sigma_k^2}\right)\right\}=0 \tag{4}$$

by (2) and (4)

$$\hat{\sigma}_k^2=\frac{\sum_{i=1}^{n}\Lambda_{i,k}\|x_i-\hat{\mu}_k\|^2}{\sum_{i=1}^{n}\Lambda_{i,k}}$$

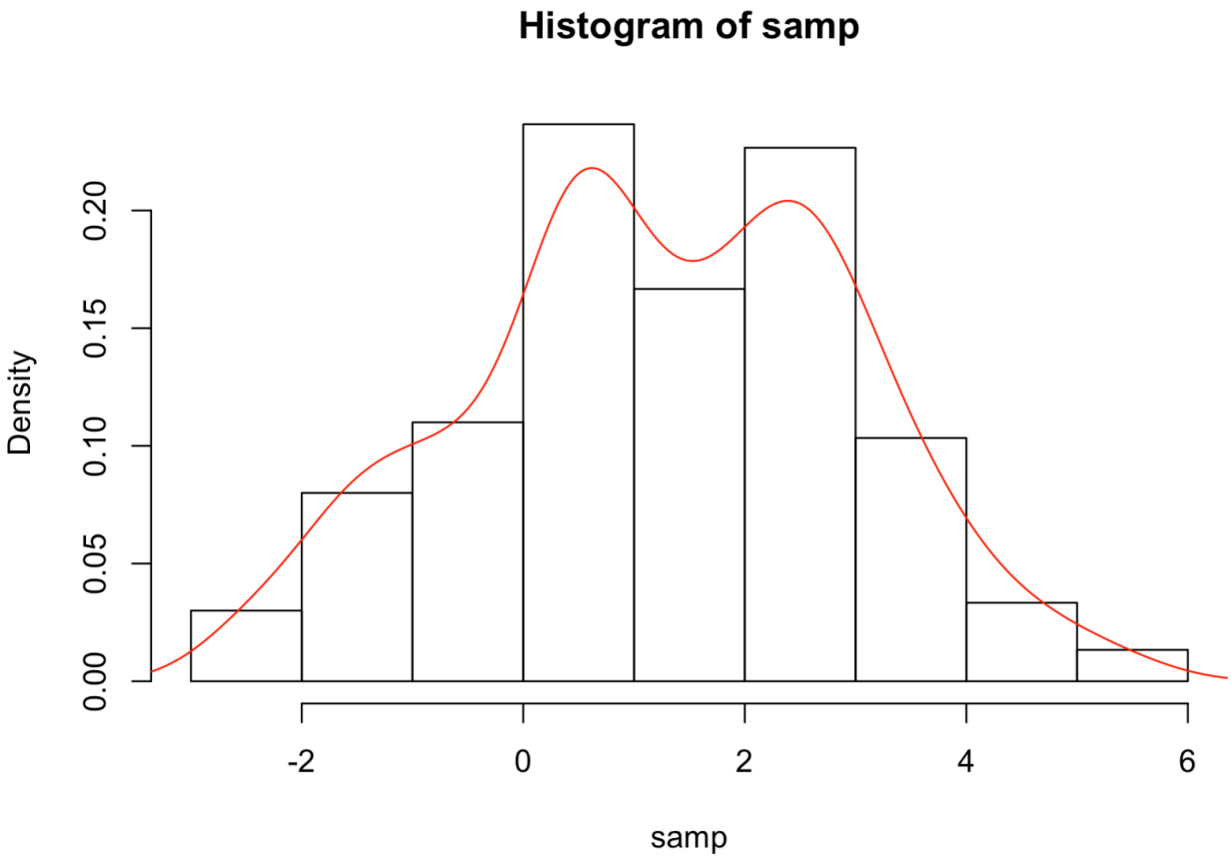
According to the definition of $\Lambda_{i,k}$, the estimate of α_k is obviously that

$$\hat{\alpha}_k=\frac{1}{n}\sum_{i=1}^{n}\Lambda_{i,k}$$

Q.E.D

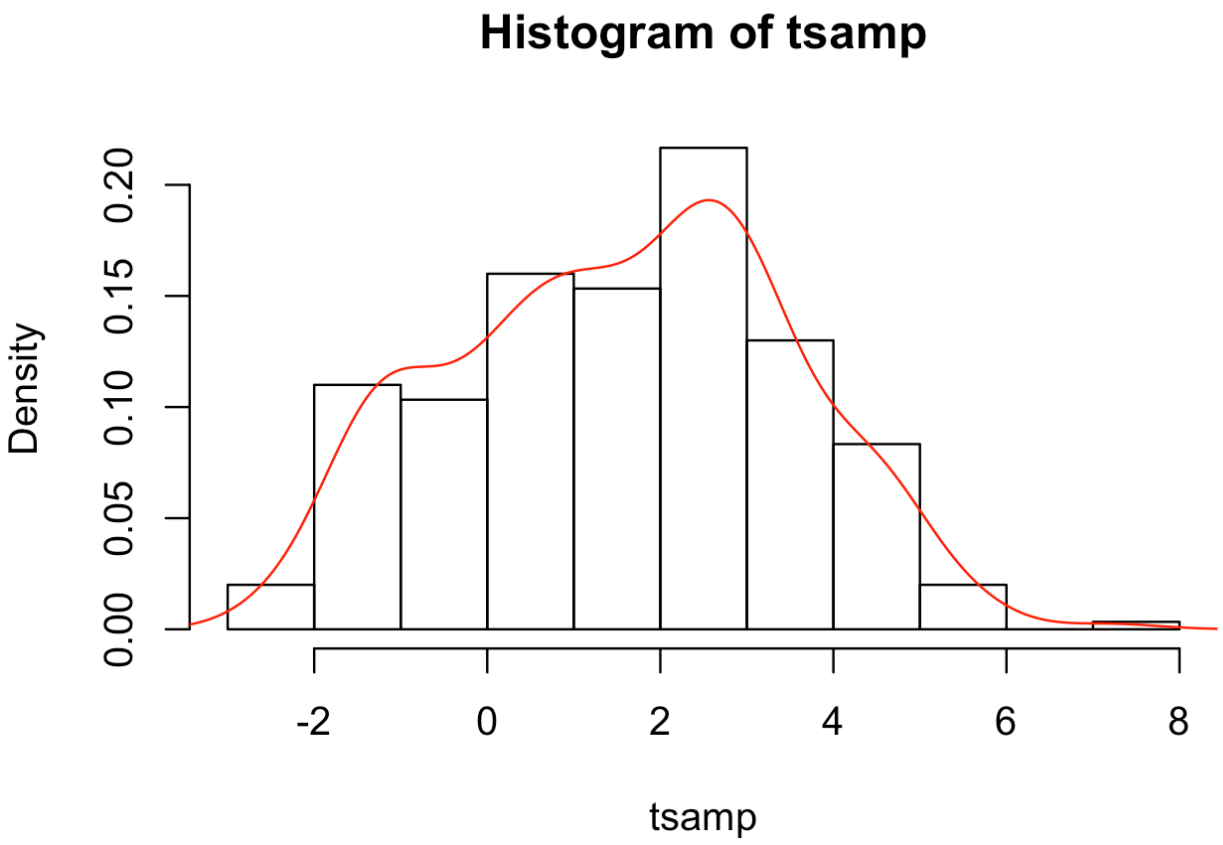
Prob 2 . For $k=3$, simulate 100 data points from such a GMM, and use the EM algorithm to estimate the parameters. Check how well the estimates are.

```
set.seed(1234)
n <- 300
alpha1 <- 0.4
mu1   <- 3
sigma1 <- 1
alpha2 <- 0.4
mu2   <- -2
sigma2 <- 2
alpha3 <- 0.2
mu3   <- 0
sigma3 <- 2^0.5
n1 <- floor(n*alpha1)
n2 <- floor(n*(alpha2+alpha1))-n1
n3 <- n-n2-n1
samp <-numeric(n)
samp[1:n1] <- rnorm(n1, mu1, sigma1)
samp[(n1+1):n1+n2] <- rnorm(n2, mu2, sigma2)
samp[(n2+1):n] <- rnorm(n3, mu3, sigma3)
hist(samp, freq = FALSE)
lines(density(samp), col = 'red')
#EM
library(mclust)
em <- Mclust(samp)
summary(em, parameters = T)
```




```
Gaussian finite mixture model fitted by EM algorithm
----------------------------------------------------

Mclust V (univariate, unequal variance) model with 3 components:

 log-likelihood   n df       BIC       ICL
      -566.8767 300  8 -1179.384 -1242.349

Clustering table:
  1   2   3
 63  52 185

Mixing probabilities:
        1         2         3
0.2089691 0.1350616 0.6559693

Means:
         1          2          3
-1.1335438  0.4727417  2.1804613

Variances:
         1          2          3
0.60241205 0.01478454 1.57299074
```

The result is quite different from the initial set parameters. This is because the three distributions are dense and coupled together, and the estimated value may not be unique. If the distribution is sparse and n is larger, the effect is better.The following figure is a histogram drawn according to the estimated value, and you can see that the fit is well.



Prob 3 . Finish all the exercises on pages entitled "Exercise".

The law school data set law in the bootstrap package contains average LSAT and average GPA for 15 law schools. This data set is a random sample from the universe of 82 law schools in law 82 .

1. Estimate the correlation between LSAT and GPA scores, and compute the bootstrap estimate of the standard error of the sample correlation.
2. Use the boot function from package boot.

```
> print(cor(law$LSAT, law$GPA)) # sample correlation coef
[1] 0.7763745
> print(cor(law82$LSAT, law82$GPA)) # population correlation coef
[1] 0.7599979
# bootstrap estimate of standard error of
B <- 200 #number of replicates
n <- nrow(law) #sample size
R <- numeric(B) #storage for replicates
#bootstrap estimate of standard error of R
for (b in 1:B) {
```

```
#randomly select the indices
i <- sample(1:n, size = n, replace = TRUE)
LSAT <- law$LSAT[i] # i -- vector of índices
GPA <- law$GPA[i]
R[b] <- cor(LSAT, GPA)
}
#output
> print(se.R <- sd(R))
[1] 0.1297349
```

Compute the bootstrap estimation of bias for the law school sample correlation problem

```
library(bootstrap)
data(law)
theta.hat <- cor(law$LSAT, law$GPA)
# bootstrap estimate of bias
B <- 2000 # number of bootstrap replicates
n <- nrow(law) # sample size
theta.b <- numeric(B)
for (b in 1:B) { # randomly select the indices
i <- sample(1:n, size = n, replace = TRUE) # i is a vector of indices
LSAT <- law$LSAT[i]
GPA <- law$GPA[i]
theta.b[b] <- cor(LSAT, GPA)
}
bias <- mean(theta.b) - theta.hat
bias
> bias
[1] 0.007822616
```

The patch (bootstrap) data contains measurements of a certain hormone in the bloodstream of eight subjects after wearing a medical patch. The parameter of interest is

$$\theta = \frac{E(new) - E(old)}{E(old) - E(placebo)}$$

If $|\theta| \leq 0.2$, this indicates bioequivalence of the old and new patches. The statistic is $\bar{Y}/\bar{Z}$. Compute a bootstrap estimate of bias in the bioequivalence ratio statistic.

```
data(patch, package = "bootstrap")
n <- nrow(patch)
y <- patch$y
z <- patch$z
theta.hat <- mean(y) / mean(z)
```

```
> print (theta.hat)
[1] -0.0713061
theta.jack <- numeric(n)
for (i in 1:n){
theta.jack[i] <- mean(y[-i]) / mean(z[-i])
}
bias <- (n - 1) * (mean(theta.jack) - theta.hat)
> print(bias)
[1] 0.008002488
```

Compute the confidence intervals for the patch ratio statistic.

```
library(boot)       #for boot and boot.ci
    data(patch, package = "bootstrap")
    theta.boot <- function(dat, ind) {
        #function to compute the statistic
        y <- dat[ind, 1]
        z <- dat[ind, 2]
        mean(y) / mean(z)
    }
    y <- patch$y
    z <- patch$z
    dat <- cbind(y, z)
    boot.obj <- boot(dat, statistic = theta.boot, R = 2000)
    print(boot.obj)
    print(boot.ci(boot.obj,
                  type = c("basic", "norm", "perc")))
    #calculations for bootstrap confidence intervals
    alpha <- c(.025, .975)
    #normal
    print(boot.obj$t0 + qnorm(alpha) * sd(boot.obj$t))
    #basic
    print(2*boot.obj$t0 -   quantile(boot.obj$t, rev(alpha), type=1))
    #percentile
    print(quantile(boot.obj$t, alpha, type=6))
```

```
ORDINARY NONPARAMETRIC BOOTSTRAP


Call:
boot(data = dat, statistic = theta.boot, R = 2000)


Bootstrap Statistics :
      original      bias    std. error
```

```
t1* -0.0713061 0.007706594   0.1009925

BOOTSTRAP CONFIDENCE INTERVAL CALCULATIONS
Based on 2000 bootstrap replicates

CALL :
boot.ci(boot.out = boot.obj, type = c("basic", "norm", "perc"))

Intervals :
Level      Normal              Basic              Percentile
95%   (-0.2770,  0.1189 )   (-0.3126,  0.0863 )   (-0.2289,  0.1700 )
Calculations and Intervals on Original Scale
>
>
> #calculations for bootstrap confidence intervals
> alpha <- c(.025, .975)
>
> #normal
> print(boot.obj$t0 + qnorm(alpha) * sd(boot.obj$t))
[1] -0.2692477  0.1266355
>
> #basic
> print(2*boot.obj$t0 -
+           quantile(boot.obj$t, rev(alpha), type=1))
      97.5%        2.5%
-0.31247046  0.08631008
>
> #percentile
> print(quantile(boot.obj$t, alpha, type=6))
      2.5%      97.5%
-0.2288723  0.1700097

```

Compute the confidence intervals using the boot. ci function.

```
library(boot)
    data(law, package = "bootstrap")
    boot.obj <- boot(law, R = 2000,
             statistic = function(x, i){cor(x[i,1], x[i,2])})
    print(boot.ci(boot.obj, type=c("basic","norm","perc")))
```

```
BOOTSTRAP CONFIDENCE INTERVAL CALCULATIONS
Based on 2000 bootstrap replicates

CALL :
boot.ci(boot.out = boot.obj, type = c("basic", "norm", "perc"))

Intervals :
Level      Normal              Basic              Percentile
95%   ( 0.5223,  1.0388 )   ( 0.5902,  1.0886 )   ( 0.4641,  0.9625 )
Calculations and Intervals on Original Scale
```

Use linear regression model to predict cat heart weights with cat body weights. Compute the confidence intervals of the regression coefficients with bootstrap resampling.

```
library(MASS)
data(cats)
resample <- function(x) {
sample(x,size=length(x),replace=TRUE)
}
coefs.cats.lm <- function(subset) {
fit <- lm(Hwt~Bwt,data=cats,subset=subset)
return(coefficients(fit))
}
cats.lm.sampling.dist <- replicate(1000,
coefs.cats.lm(resample(1:nrow(cats))))
(limits <- apply(cats.lm.sampling.dist,1,quantile,c(0.025,0.975)))
```

```
      (Intercept)      Bwt
2.5%    -1.948358 3.442391
97.5%    1.210477 4.636989
```