University of Edinburgh, School of Mathematics
Incomplete Data Analysis, 2020/2021
EM algorithm and the bootstrap

Vanda Inácio

So far we have been applying the EM algorithm to obtain maximum likelihood estimates for a broad type of incomplete data problems. However, the standard error estimates are not readily available at the conclusion of the algorithm as they would be, for example, from direct maximisation of the observed data likelihood using optimisation techniques, such as the Newton–Raphson algorithm, in which the observed information matrix is calculated at each iteration. Although there are analytical methods available (e.g., Louis, 1982), we discuss a general approach to obtain standard error estimates/confidence intervals: the bootstrap (Efron, 1979).

The essential concept of bootstrapping is to emulate repetition of the experiment by simulating new data on the computer, followed by recalculation of the the estimate of interest using the simulated data. The bootstrap effectively replaces the calculus and theory (to obtain standard errors/confidence intervals) with pure computational effort. But, of course, does not eliminate the need of thinking! The term 'bootstrap' was chosen by Efron (1979) in the first comprehensive account of this computer-intensive methodology.

The bootstrap emulates the sampling distribution of $\widehat{\theta}$ by emulating the processes of data generation and model fitting. It does this by generating artificial data, say $y^{(b)} = (y_1^{(b)}, \ldots, y_n^{(b)})$, from a distribution that approximates the true unknown sampling distribution of the actual data, followed by recalculating the estimate using the artificial data. This is done a large number of times, say B, resulting in a large collection of bootstrap estimates, denoted by $\widehat{\theta}^{(b)}$, $b = 1, \ldots, B$. The distribution of these artificially generated bootstrap estimates can then be used to infer the sampling distribution of $\widehat{\theta}$.

In nonparametric bootstrapping, the empirical cumulative distribution function (ecdf)

$$\widehat{F}_{\text{ecdf}}(y) = \frac{1}{n}\sum_{i=1}^{n} I(y_i \leq y)$$

can be used as a discrete approximation to the true unknown cumulative distribution function. Note that the ecdf assigns probability mass $1/n$ to each y_i value. The nonparametric bootstrap generates new data $y^{(b)}$ by random sampling from the ecdf. Note that random sampling from the ecdf can be accomplished by sampling with replacement from the observed data $y_1, \ldots, y_n$, and in `R` this is accomplished through the use of the function `sample`.

The general algorithm for nonparametric bootstrapping is as follows. For $b = 1, \ldots, B$:

1. Generate a bootstrap sample $y^{(b)} = (y_1^{(b)}, \ldots, y_n^{(b)})$ by randomly sampling from $(y_1, \ldots, y_n)$ with replacement.
2. Compute $\widehat{\theta}^{(b)}$ using $y^{(b)} = (y_1^{(b)}, \ldots, y_n^{(b)})$.

The bootstrap variance can be calculated as

$$\text{var}_{\text{boot}}(\widehat{\theta}) = \frac{1}{B-1}\sum_{b=1}^{B}(\widehat{\theta}^{(b)} - \widehat{\theta}_{\text{boot}})^2,$$

where $\widehat{\theta}_{\text{boot}} = \frac{1}{B}\sum_{b=1}^{B}\widehat{\theta}^{(b)}$.

There are several ways to compute bootstrap confidence intervals, we restrict ourselves to the most commonly used form of bootstrap confidence intervals, the percentile method. This method is the simplest and it performs well in most situations. Under this method, a $100(1-\alpha)\%$ bootstrap confidence interval is computed as

$$(\widehat{\theta}^{(b,l)}, \widehat{\theta}^{(b,u)}),$$

where $\widehat{\theta}^{(b,l)}$ and $\widehat{\theta}^{(b,u)}$ are, respectively, the empirical $\alpha/2$ and $1-\alpha/2$ quantiles of the collection

$$(\widehat{\theta}^{(1)}, \ldots, \widehat{\theta}^{(B)}).$$

The question that now arises is: "How many bootstrap simulations should we perform?" The number of bootstrap simulations B must be large enough that the estimated standard errors or quantiles are not materially affected by the inherent randomness of the bootstrap. That is, they should be negligibly if the bootstrap procedure were to be repeated. Davison and Hinkley (1997) recommended performing at least 1000 bootstrap simulations when calculating 95% bootstrap confidence intervals.

We start revisiting the code implementing the EM algorithm for the genetic linkage example. We had $y = (125, 18, 20, 34)$ and the E-step was given by

$$Q(\theta \mid \theta^{(t)}) = (y_1 - z^{(t)} + y_4)\log\theta + (y_2 + y_3)\log(1-\theta),$$
$$z^{(t)} = y_1 \times \frac{1/2}{1/2 + \theta^{(t)}/4},$$

while the M-step was

$$\theta^{(t+1)} = \frac{y_1 - z^{(t)} + y_4}{n - z^{(t)}}, \quad n = y_1 + y_2 + y_3 + y_4.$$

```
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)
}

y <- c(125, 18, 20, 34)
multi(y = y, 0.5, 0.00001)
```

```
## [1] 0.6268207
```

In order to resample from our original data, we have to re-write it in an alternative way.

```
y <- c(rep(1, 125), rep(2, 18), rep(3, 20), rep(4, 34))
n <- length(y)
```

```
B <- 2000
thetahatb <- numeric(B)
set.seed(1)
for(l in 1:B){
yb <- sample(y, size = n, replace = T)
thetahatb[l] <- multi(as.numeric(table(yb)), 0.1, 0.00001)
}

sd(thetahatb)
```

```
## [1] 0.05206568
```

```
quantile(thetahatb, c(0.025, 0.975))
```

```
##      2.5%     97.5%
## 0.5223200 0.7308859
```

We will now implement the (nonparametric) bootstrap approach for the two-component Gaussian mixture model. As in the previous case, we start by revisiting the code that implements the EM algorithm and then we implement a nonparametric bootstrap in a similar fashion to what we have done in the previous example.

```
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)
}
```

```r
data(faithful)
attach(faithful)

res <- em.mixture.two.normal(y = waiting, c(0.4, 45, 4, 90, 4), 0.00001)
p <- res[1]
mu1 <- res[2]
sigma1 <- res[3]
mu2 <- res[4]
sigma2 <- res[5]

B <- 2000
y <- waiting
n <- length(y)
estimates <- matrix(0, nrow = 5, ncol = B)
set.seed(1)
for(l in 1:B){
yb <- sample(y, size = n, replace = T)
estimates[,l] <- em.mixture.two.normal(yb, c(0.4, 45, 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,])

df <- data.frame("Estimate" = c(p, mu1, mu2, sigma1, sigma2),
                 "lq" = c(pl, mu1l, mu2l, sigma1l, sigma2l),
                 "uq" = c(ph, mu1h, mu2h, sigma1h, sigma2h))

rownames(df) <- c("$p$", "$\\mu_1$","$\\mu_2$","$\\sigma_1$", "$\\sigma_2$")
colnames(df) <- c("Estimate", "2.5% quantile", "97.5% quantile")
knitr::kable(df, escape = FALSE, digits = 3, )
```

	Estimate	2.5% quantile	97.5% quantile
p	0.361	0.301	0.424
μ_1	54.615	53.059	56.132
μ_2	80.091	79.043	81.073
σ_1	5.871	4.889	6.782
σ_2	5.868	5.067	6.783