STAT 600: Homework 3

Hyungjoon Kim, Colorado State University

February 27, 2024

Problem 1

a) Let $\delta \sim \text{Bernoulli}\,(p)$, then we can write the joint distribution of Y and δ as

$$f_{Y,\delta}\left(y_i, \delta_i \mid \boldsymbol{\theta}\right) = \left[pf\left(y_i \mid \lambda\right)\right]^{\delta_i}\left[\left(1-p\right)f\left(y_i \mid \mu\right)\right]^{1-\delta_i}$$

So, the complete log-likelihood function of $\boldsymbol{\theta} = (\lambda, \mu, p)$ is

$$\ell\left(\boldsymbol{\theta}\right) = \sum_{i=1}^{n}\left[\delta_i \log f\left(y_i \mid \lambda\right) + \delta_i \log p + \left(1-\delta_i\right)\log f\left(y_i \mid \mu\right) + \left(1-\delta_i\right)\log\left(1-p\right)\right]$$

Then, we can find that the conditional distribution of δ given Y is

$$\begin{aligned} f_{\delta|Y}\left(\delta_i \mid y_i\right) &= \frac{f_{Y,\delta}\left(y_i, \delta_i \mid \boldsymbol{\theta}\right)}{f_Y\left(y_i \mid \boldsymbol{\theta}\right)} \\ &= \frac{\left[pf\left(y_i \mid \lambda\right)\right]^{\delta_i}\left[\left(1-p\right)f\left(y_i \mid \mu\right)\right]^{1-\delta_i}}{pf\left(y_i \mid \lambda\right) + \left(1-p\right)f\left(y_i \mid \mu\right)} \end{aligned}$$

So, $\delta_i \mid Y_i \sim \text{Bernoulli}\,(w_i)$, where

$$w_i = \frac{pf\left(y_i \mid \lambda\right)}{pf\left(y_i \mid \lambda\right) + \left(1-p\right)f\left(y_i \mid \mu\right)}$$

Thus, consider $\boldsymbol{\theta}^{(v)} = \left(\lambda^{(v)}, \mu^{(v)}, p^{(v)}\right)$ be an update of vth iteration, then we can obtain $w_i^{(v)}$ by calculating

$$w_i^{(v)} = \frac{p^{(v)} f\left(y_i \mid \lambda^{(v)}\right)}{p^{(v)} f\left(y_i \mid \lambda^{(v)}\right) + \left(1-p^{(v)}\right)f\left(y_i \mid \mu^{(v)}\right)}$$

And we can write the Q function, the expectation of the complete log-likelihood based on $\boldsymbol{\theta}^{(v)}$, as

$$\begin{aligned} Q\left(\boldsymbol{\theta}, \boldsymbol{\theta}^{(v)}, \boldsymbol{y}\right) &= \mathbb{E}_{\boldsymbol{\theta}^{(v)}}\left[\ell\left(\boldsymbol{\theta}\right)\right] \\ &= \sum_{i=1}^{n}\left[w_i^{(v)}\log f\left(y_i \mid \lambda^{(v)}\right) + \left(1-w_i^{(v)}\right)\log f\left(y_i \mid \mu^{(v)}\right) + w_i^{(v)}\log p^{(v)} + \left(1-w_i^{(v)}\right)\log\left(1-p^{(v)}\right)\right] \end{aligned}$$

Since $\log f\left(y_i \mid \lambda\right) = \log\lambda - \lambda y_i$,

$$\begin{aligned} Q\left(\boldsymbol{\theta}, \boldsymbol{\theta}^{(v)}, \boldsymbol{y}\right) = &\sum_{i=1}^{n} w_i^{(v)}\left(\log\lambda^{(v)} - \lambda^{(v)} y_i\right) \\ &+ \sum_{i=1}^{n}\left(1-w_i^{(v)}\right)\left(\log\mu^{(v)} - \mu^{(v)} y_i\right) \\ &+ \sum_{i=1}^{n}\left[w_i^{(v)}\log p^{(v)} + \left(1-w_i^{(v)}\right)\log\left(1-p^{(v)}\right)\right] \end{aligned}$$

b) Note that

$$\frac{\partial}{\partial \lambda^{(v)}} Q\left(\boldsymbol{\theta}, \boldsymbol{\theta}^{(v)}, \boldsymbol{y}\right) = \sum_{i=1}^{n} w_i^{(v)} \left(\frac{1}{\lambda^{(v)}} - y_i\right) \overset{\text{set}}{=} 0$$
$$\frac{\partial}{\partial \mu^{(v)}} Q\left(\boldsymbol{\theta}, \boldsymbol{\theta}^{(v)}, \boldsymbol{y}\right) = \sum_{i=1}^{n} \left(1 - w_i^{(v)}\right) \left(\frac{1}{\mu^{(v)}} - y_i\right) \overset{\text{set}}{=} 0$$
$$\frac{\partial}{\partial p^{(v)}} Q\left(\boldsymbol{\theta}, \boldsymbol{\theta}^{(v)}, \boldsymbol{y}\right) = \sum_{i=1}^{n} \left(\frac{w_i^{(v)}}{p^{(v)}} - \frac{1 - w_i^{(v)}}{1 - p^{(v)}}\right) \overset{\text{set}}{=} 0$$

Solving the equations above, we can obtain

$$\lambda^{(v+1)} = \frac{\sum_{i=1}^{n} w_i^{(v)}}{\sum_{i=1}^{n} w_i^{(v)} y_i}$$
$$\mu^{(v+1)} = \frac{\sum_{i=1}^{n} \left(1 - w_i^{(v)}\right)}{\sum_{i=1}^{n} \left(1 - w_i^{(v)}\right) y_i}$$
$$p^{(v+1)} = \frac{\sum_{i=1}^{n} w_i^{(v)}}{n}$$

Therefore, starting with $\lambda^{(0)}$, $\mu^{(0)}$, and $p^{(0)}$, we can update each value iteratively using formulae above.

Problem 2

I attach a function `exponential_mixture_EM` in `Rcpp` below:

```
#include <RcppArmadillo.h>
// [[Rcpp::depends(RcppArmadillo)]]

using namespace Rcpp;

// [[Rcpp::export]]
NumericVector exponential_mixture_EM(NumericVector y,
                                     NumericVector inits,
                                     double tol = 1e-4,
                                     int max_iter = 1000){
  double n = (double)y.size();
  double err = 0.0;

  NumericVector curr(3);
  NumericVector new_val(3);

  NumericVector a(n, 0);
  NumericVector b(n, 0);
  NumericVector w(n, 0);

  curr = inits;
  for(int i = 0; i < max_iter; i++){
    a = curr(0) * dexp(y, 1.0 / curr(1));
    b = (1.0 - curr(0)) * dexp(y, 1.0 / curr(2));
    w = a / (a + b);

    new_val(0) = sum(w) / n;
    new_val(1) = sum(w) / sum(w * y);
```

```
    new_val(2) = sum(1.0 - w) / sum((1.0 - w) * y);

    err = sqrt(sum(pow(new_val - curr, 2.0)));

    curr = new_val;

    if(err < tol){
      break;
    }
  }

  return(curr);
}
```

Problem 3

I generated $n = 100$ observations from the true distribution with $\boldsymbol{\theta} = (p, \lambda, \mu) = (0.25, 1, 2)$ 100 times and stored them in a list y. To make it reproducible, I set a seed number.

```
generate_data <- function(n, p, lambda, mu){
  y <- numeric(n)
  for(i in 1:n){
    if(runif(1) < p){
      y[i] <- rexp(1, rate = lambda)
    }else{
      y[i] <- rexp(1, rate = mu)
    }
  }
  return(y)
}

set.seed(2024)

n <- 100
p <- 0.25
lambda <- 1
mu <- 2

nsim <- 100
y <- list()
for(i in 1:nsim){
  y[[i]] <- generate_data(n, p, lambda, mu)
}
```

Problem 4

Using EM algorithm, I estimated p, λ, and μ 100 times using 100 simulated datasets. The average of estimates are 0.1997, 1.2992, and 1.7199 respectively.

```
res <- data.frame(matrix(nrow = nsim, ncol = 3))
colnames(res) <- c("p", "lambda", "mu")
for(i in 1:nsim){
  res[i, ] <- exponential_mixture_EM(y[[i]],
```

```
                                    c(0.2, 0.5, 1.5))
}

res <- rbind(round(colMeans(res), 4),
             round(apply(res, 2, sd), 4))
rownames(res) <- c("Mean", "StDev")
res
```

```
##            p lambda     mu
## Mean  0.1997 1.2992 1.7199
## StDev 0.0071 0.2323 0.1565
```

Problem 5

I used bootstrap to estimate the standard errors of the parameter estimates. Using the last synthetic data I created in the problem 2, I took a sample of size 100 with replacement, and estimated parameters using EM algorithm. The number of bootstrap samples are $10,000$, so I obtained $10,000$ sets of parameters, and calculated standard errors using those parameter estimates. The bootstrap standard errors of $\hat{p}$, $\hat{\lambda}$, and $\hat{\mu}$ are 0.003, 0.1865, and 0.1364 respectively.

```
y100 <- y[[100]]
res_y100 <- exponential_mixture_EM(y100, c(0.2, 0.5, 1.5))

res_boot <- data.frame(matrix(nrow = 10000, ncol = 3))
colnames(res_boot) <- c("p", "lambda", "mu")
for(i in 1:10000){
  y_bootstrap <- sample(y100, n, replace = TRUE)
  res_boot[i, ] <- exponential_mixture_EM(y_bootstrap, res_y100)
}
res_boot <- rbind(round(apply(res_boot, 2, mean), 4),
                  round(sqrt(apply(res_boot, 2, var) * 9999 / 10000), 4))
rownames(res_boot) <- c("EM Estimates", "SE (Bootstrap)")
res_boot
```

```
##                     p lambda     mu
## EM Estimates   0.2011 1.1629 1.4015
## SE (Bootstrap) 0.0030 0.1865 0.1364
```

Problem 6

Using 100 samples generated in the problem 3, I calculated EM estimates and bootstrap standard error for each sample. As we know the parameter, $(0.25, 1, 2)$, I calculated bias by subtracting the parameter vector from the estimates vector. To obtain 95 confidence intervals for the parameters, I constructed percentile bootstrap confidence intervals using the same bootstrap estimates which are used to calculate bootstrap standard errors. Then using such confidence intervals, I checked whether a confidence interval captures the parameter or not. The average estimates, bias, and standard error is attached below, and I could find that there is a systematic bias of each estimate, and in my opinion, it caused low coverage rates of confidence intervals.

```
nboot <- 10000
params <- c(p, lambda, mu)
estimates <- data.frame(matrix(nrow = nsim, ncol = 3))
ses <- data.frame(matrix(nrow = nsim, ncol = 3))
res_boot <- data.frame(matrix(nrow = nboot, ncol = 3))
```

```r
coverage <- data.frame(matrix(nrow = nsim, ncol = 3))
for(i in 1:nsim){
  estimates_each <- exponential_mixture_EM(y[[i]], c(0.2, 0.5, 1.5))
  for(b in 1:nboot){
    y_bootstrap <- sample(y[[i]], n, replace = TRUE)
    res_boot[b, ] <- exponential_mixture_EM(y_bootstrap, estimates_each)
  }
  estimates[i, ] <- estimates_each
  ses[i, ] <- sqrt(apply(res_boot, 2, var) * (nboot - 1) / nboot)
  for(l in 1:3){
    ci <- quantile(res_boot[, l], probs = c(0.025, 0.975))
    coverage[i, l] <- (ci[1] <= params[l]) & (ci[2] >= params[l])
  }
}
bias <- estimates - matrix(rep(c(p, lambda, mu), nsim),
                           nrow = nsim, byrow = TRUE)

res6 <- rbind(apply(estimates, 2, mean),
              apply(bias, 2, mean),
              apply(ses, 2, mean),
              apply(coverage, 2, mean))
rownames(res6) <- c("Estimate", "Bias", "SE", "Coverage")
colnames(res6) <- c("$$p$$", "$$\\lambda$$", "$$\\mu$$")
knitr::kable(round(res6, 4), caption = "Simulation Results")
```

Table 1: Simulation Results

	p	λ	μ
Estimate	0.1997	1.2992	1.7199
Bias	-0.0503	0.2992	-0.2801
SE	0.0029	0.2030	0.1706
Coverage	0.0000	0.2300	0.6100