STAT 600 - HW 3

Kevin Korsurat

All Rcpp/RcppArmadillo can be found in my GitHub.

Derivation

First, I will derive the log-likelihood function given the completed data, $l\left(p, \lambda, \mu | \boldsymbol{Y}, \boldsymbol{\delta}\right)$.

$$\begin{aligned}
l\left(p, \lambda, \mu | \boldsymbol{Y}, \boldsymbol{\delta}\right) &= \log\left(L\left(p, \lambda, \mu | \boldsymbol{Y}, \boldsymbol{\delta}\right)\right) \\
&= \log\left(\prod_{i=1}^{n} p\left(Y_i, \delta_i | p, \lambda, \mu\right)\right) \\
&= \log\left(\prod_{i=1}^{n} \left(p\lambda \exp\left(-\lambda y_i\right)\right)^{\delta_i} \left(\left(1-p\right)\mu \exp\left(-\mu y_i\right)\right)^{1-\delta_i}\right) \\
&= \sum_{i=1}^{n} \log\left\{\left(p\lambda \exp\left(-\lambda y_i\right)\right)^{\delta_i} \left(\left(1-p\right)\mu \exp\left(-\mu y_i\right)\right)^{1-\delta_i}\right\} \\
&= \sum_{i=1}^{n} \left[\delta_i \left\{-\lambda y_i + \log\left(p\lambda\right)\right\} + \left(1-\delta_i\right)\left\{-\mu y_i + \log\left(\left(1-p\right)\mu\right)\right\}\right]
\end{aligned}$$

Then, the E-step in the EM algorithm is shown below.

$$\begin{aligned}
Q\left(p, \lambda, \mu | p^{(t)}, \lambda^{(t)}, \mu^{(t)}\right) &= E\left[l\left(p, \lambda, \mu | \boldsymbol{Y}, \boldsymbol{\delta}\right) | \boldsymbol{Y}, p^{(t)}, \lambda^{(t)}, \mu^{(t)}\right] \\
&= E\left[\sum_{i=1}^{n} \left[\delta_i \left\{-\lambda y_i + \log\left(p\lambda\right)\right\} + \left(1-\delta_i\right)\left\{-\mu y_i + \log\left(\left(1-p\right)\mu\right)\right\}\right] \middle| \boldsymbol{Y}, p^{(t)}, \lambda^{(t)}, \mu^{(t)}\right] \\
&= \sum_{i=1}^{n} E\left[\delta_i \left\{-\lambda y_i + \log\left(p\lambda\right)\right\} + \left(1-\delta_i\right)\left\{-\mu y_i + \log\left(\left(1-p\right)\mu\right)\right\} \middle| y_i, p^{(t)}, \lambda^{(t)}, \mu^{(t)}\right] \\
&= \sum_{i=1}^{n} \left\{\hat{\delta}_i^{(t)} \left[-\lambda y_i + \log\left(p\lambda\right)\right] + \left(1-\hat{\delta}_i^{(t)}\right)\left[-\mu y_i + \log\left(\left(1-p\right)\mu\right)\right]\right\}
\end{aligned}$$

where $\hat{\delta}_i^{(t)} = E\left[\delta_i | \, y_i, p^{(t)}, \lambda^{(t)}, \mu^{(t)}\right] = \frac{p^{(t)}\lambda^{(t)} \exp\left(-\lambda^{(t)} y_i\right)}{p^{(t)}\lambda^{(t)} \exp\left(-\lambda^{(t)} y_i\right) + \left(1-p^{(t)}\right)\mu^{(t)} \exp\left(-\mu^{(t)} y_i\right)}$

Then, I will derive the M-step by starting the finding the $p^{(t+1)}$.

$$
\begin{aligned}
\frac{d}{dp}Q\left(p,\lambda,\mu|p^{(t)},\lambda^{(t)},\mu^{(t)}\right) &\overset{\text{set}}{=} 0 \\
\sum_{i=1}^{n}\left[\frac{\hat{\delta}_i^{(t)}}{p}-\frac{1-\hat{\delta}_i^{(t)}}{1-p}\right] &= 0 \\
\sum_{i=1}^{n}\left[(1-p)\hat{\delta}_i^{(t)}-\left(1-\hat{\delta}_i^{(t)}\right)p\right] &= 0 \\
\sum_{i=1}^{n}\hat{\delta}_i^{(t)}-np &= 0 \\
p^{(t+1)} &= \frac{1}{n}\sum_{i=1}^{n}\hat{\delta}_i^{(t)}
\end{aligned}
$$

Below is the derivation for $\lambda^{(t+1)}$.

$$
\begin{aligned}
\frac{d}{d\lambda}Q\left(p,\lambda,\mu|p^{(t)},\lambda^{(t)},\mu^{(t)}\right) &\overset{\text{set}}{=} 0 \\
\sum_{i=1}^{n}\left[-\hat{\delta}_i^{(t)}y_i+\frac{\hat{\delta}_i^{(t)}}{\lambda}\right] &= 0 \\
-\lambda\sum_{i=1}^{n}\hat{\delta}_i^{(t)}y_i+\sum_{i=1}^{n}\hat{\delta}_i^{(t)} &= 0 \\
\lambda^{(t+1)} &= \frac{\sum_{i=1}^{n}\hat{\delta}_i^{(t)}}{\sum_{i=1}^{n}\hat{\delta}_i^{(t)}y_i}
\end{aligned}
$$

Lastly, this is the derivation for $\mu^{(t+1)}$

$$
\begin{aligned}
\frac{d}{d\mu}Q\left(p,\lambda,\mu|p^{(t)},\lambda^{(t)},\mu^{(t)}\right) &\overset{\text{set}}{=} 0 \\
\sum_{i=1}^{n}\left[\left(1-\hat{\delta}_i^{(t)}\right)y_i+\frac{1-\hat{\delta}_i^{(t)}}{\mu}\right] &= 0 \\
-\mu\sum_{i=1}^{n}\left(1-\hat{\delta}_i^{(t)}\right)y_i+\sum_{i=1}^{n}\left(1-\hat{\delta}_i^{(t)}\right) &= 0 \\
\mu^{(t+1)} &= \frac{\sum_{i=1}^{n}\left(1-\hat{\delta}_i^{(t)}\right)}{\sum_{i=1}^{n}\left(1-\hat{\delta}_i^{(t)}\right)y_i}
\end{aligned}
$$

Note: Louis's Method

According to equations (4.29) and (4.32), we can conclude that the Hessian matrix for $\hat{\boldsymbol{\theta}}=\left(\hat{p},\hat{\lambda},\lambda\mu\right)$ can be calculated by $Q''\left(\boldsymbol{\theta}\middle|\boldsymbol{\omega}\right)\Big|_{\boldsymbol{\omega}=\boldsymbol{\theta}}-l\left(\boldsymbol{\theta}\middle|\boldsymbol{y}\right)$. The variance can be calculated by taking the diagonal of the inverse of this calculated matrix. Since we are interested only in the variance of the parameters, we can consider only the diagonal elements. The derivative can be seen in the Rcpp code (I use an R function to calculate the derivative).

Result

We begin by running the EM algorithm. I have set the starting values as $\boldsymbol{\theta}0 = (p = 0.25, \lambda = 0.5, \mu = 2.5)$. The stopping criterion for the EM algorithm is $||\boldsymbol{\theta}^{(t+1)} - \boldsymbol{\theta}^{(t)}||2 < 1 \times 10^{-5}$, where $||\cdot||_2$ denotes the Euclidean distance.

Table 1 shows the estimates and the bias for each estimate. We might notice that 'p' and 'λ' tend to be underestimated when the model is overestimating 'μ'. I believe this is reasonable as the starting point for 'p' and 'λ' is lower than the actual value, while the starting point for 'μ' is greater than the correct value. We might also notice that the model tries to reach the actual values, but it still cannot perform well. Another finding that I did not show in this report is that this model is sensitive to the starting point. Additionally, Figure 1 also visualizes the distribution from the EM algorithm for each parameter. We notice that the results are pretty close to the actual values.

Table 1: The average esitmate and bias for each parameter

	p	λ	μ
Estimate	0.18921 (SD = 0.02141)	0.77191 (SD = 0.12918)	2.21487 (SD = 0.19943)
Bias	-0.06079 (SD = 0.02141)	-0.22809 (SD = 0.12918)	0.21487 (SD = 0.19943)

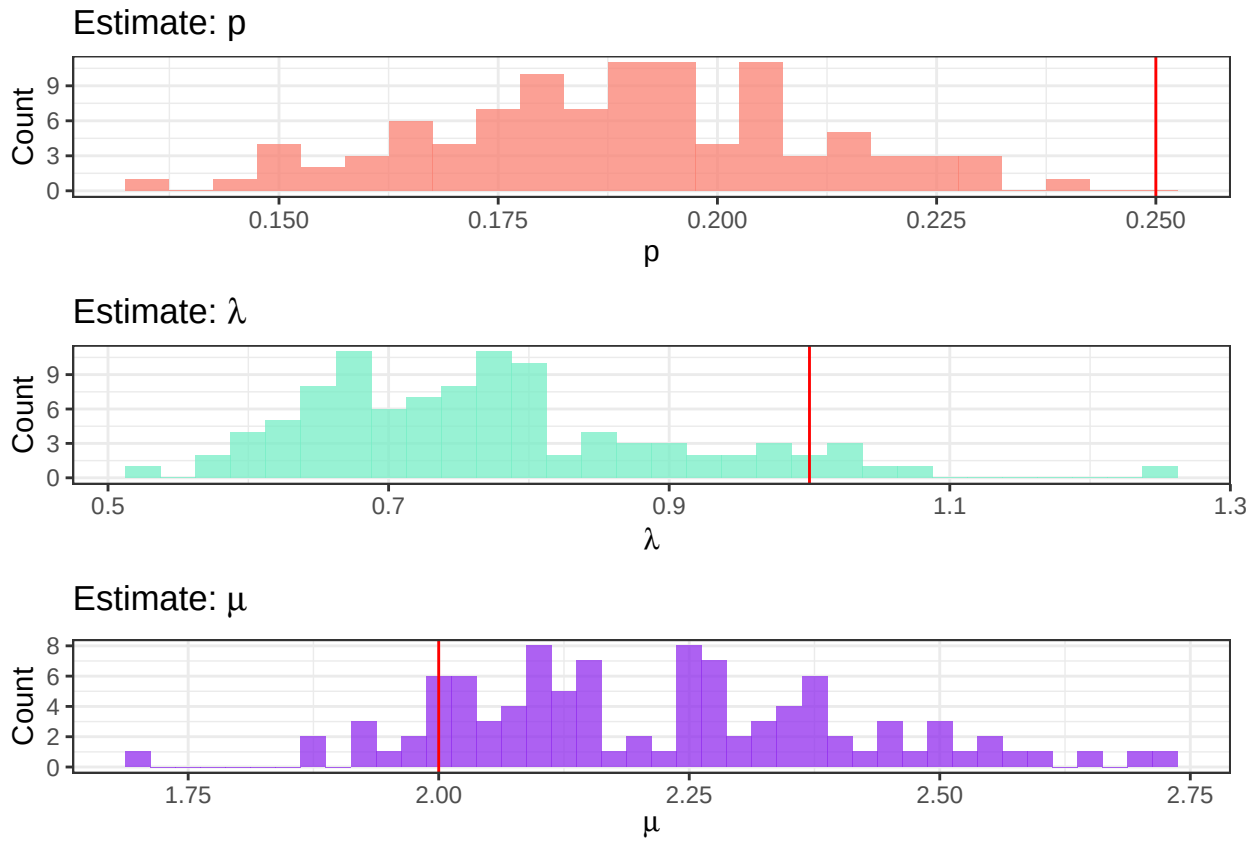

Figure 1: The distribution of the result from EM for each parameter.

Tables 2 and 3 show the standard error calculations based on Louis's method and the bootstrap, respectively. Note that for the bootstrap method, I have set M (the number of resample sampling) = 1000. We notice that Louis's method provided the higher standard error compared to the bootstrap. This leads to the conclusion

that the coverage probability provided by Louis's method is higher than the one with the bootstrap. Note that the 95% confidence interval is calculated as $\hat{\theta}_j \pm z_{0.05/2}\text{SE}\left(\hat{\theta}_j\right)$.

Table 2: The standard error for each parameter by using Louis's Method

	p	λ	μ
Standard Error	0.0431938	0.3508719	0.3967152
Coverage Probability	82.0000000	99.0000000	100.0000000

Table 3: The standard error for each parameter by using the bootstrap Method

	p	λ	μ
Standard Error	0.0205225	0.1151702	0.1848338
Coverage Probability	20.0000000	41.0000000	79.0000000

Appendix

```
knitr::opts_chunk$set(echo = FALSE)

library(tidyverse)
library(knitr)
library(Rcpp)
library(RcppArmadillo)
library(foreach)
library(doParallel)
library(ggplot2)
library(latex2exp)
library(gridExtra)

path <- "/Users/kevin-imac/Desktop/Github - Repo/"
if(! file.exists(path)){
  path <- "/Users/kevinkvp/Desktop/Github Repo/"
}

sourceCpp(paste0(path, "HW3EM/src/main.cpp"))

### User-defined functions ----------------------------------------------------
meanSD <- function(x, dplace = 5){
  mm <- round(mean(x), digits = dplace)
  ss <- round(sd(x), digits = dplace)
  paste0(mm, " (SD = ", ss, ")")
}

### Simulated the data
set.seed(31082, kind = "L'Ecuyer-CMRG")
registerDoParallel(5)
simDat <- foreach(t = 1:100) %dopar% {

  ### Simulate the data
  clus_ind <- rbinom(100, 1, 0.25)
  y <- rexp(100, rate = ifelse(clus_ind == 1, 1, 2))

  y

}
stopImplicitCluster()

### Run the model
set.seed(31082, kind = "L'Ecuyer-CMRG")
registerDoParallel(5)
resultEM <- foreach(t = 1:100, .combine = "rbind") %dopar% {

  em_result <- EM_rcpp(y = simDat[[t]], p0 = 0.2, lambda0 = 0.5, mu0 = 2.5, eps = 1e-5)
  c(em_result$p, em_result$lambda, em_result$mu)

}
stopImplicitCluster()
```

```r
### Louis SE
set.seed(31082, kind = "L'Ecuyer-CMRG")
registerDoParallel(5)
VARLouis <- foreach(t = 1:100, .combine = "rbind") %dopar% {

  varMat <- iY(y = simDat[[t]], p = resultEM[t, 1], lb = resultEM[t, 2], mu = resultEM[t, 3]) -
    iX(y = simDat[[t]], p = resultEM[t, 1], lb = resultEM[t, 2], mu = resultEM[t, 3])
  1/(diag(varMat))

}
stopImplicitCluster()

#### Bootstrap
set.seed(31082, kind = "L'Ecuyer-CMRG")
registerDoParallel(5)
VARBoots <- foreach(t = 1:100, .combine = "rbind") %dopar% {

  bootMat <- BootResult(y = simDat[[t]], p0 = 0.2, lambda0 = 0.5, mu0 = 2.5,
                        eps = 1e-5, EMinit = resultEM[t, ], M = 1000)
  apply(bootMat, 2, var)

}
stopImplicitCluster()

### Table Summary: Estimate and Bias
data.frame(est = apply(resultEM, 2, meanSD),
           bias = apply(resultEM - matrix(c(0.25, 1, 2), ncol = 3, nrow = 100, byrow = TRUE), 2, meanSD)
  t() %>%
  `rownames<-`(c("Estimate", "Bias")) %>%
  kable(col.names = c("p", "$\\lambda$", "$\\mu$"),
        caption = "The average esitmate and bias for each parameter")

### Distribution of the result
plot0 <- ggplot(data.frame(x = resultEM[, 1]), aes(x = x)) +
  geom_histogram(fill = "salmon", alpha = 0.75, binwidth = 0.005) +
  theme_bw() +
  geom_vline(xintercept = 0.25, color = "red") +
  labs(x = "p", y = "Count", title = "Estimate: p")

plot1 <- ggplot(data.frame(x = resultEM[, 2]), aes(x = x)) +
  geom_histogram(fill = "aquamarine2", alpha = 0.75, binwidth = 0.025) +
  theme_bw() +
  geom_vline(xintercept = 1, color = "red") +
  labs(x = TeX("$\\lambda$"), y = "Count", title = TeX("Estimate: $\\lambda$"))

plot2 <- ggplot(data.frame(x = resultEM[, 3]), aes(x = x)) +
  geom_histogram(fill = "purple2", alpha = 0.75, binwidth = 0.025) +
  theme_bw() +
  geom_vline(xintercept = 2, color = "red") +
  labs(x = TeX("$\\mu$"), y = "Count", title = TeX("Estimate: $\\mu$"))

grid.arrange(plot0, plot1, plot2)
data.frame(sqrt(colMeans(VARLouis)),
```

```
          ((resultEM - (qnorm(1 - (0.05/2)) * sqrt(VARLouis)) <= matrix(c(0.25, 1, 2), ncol = 3, nrow =
             (matrix(c(0.25, 1, 2), ncol = 3, nrow = 100, byrow = TRUE) <= resultEM + (qnorm(1 - (0.05/
            colSums()) %>%
  t() %>%
  `rownames<-`(c("Standard Error", "Coverage Probability")) %>%
  kable(col.names = c("p", "$\\lambda$", "$\\mu$"),
        caption = "The standard error for each parameter by using Louis's Method")
data.frame(sqrt(colMeans(VARBoots)),
          ((resultEM - (qnorm(1 - (0.05/2)) * sqrt(VARBoots)) <= matrix(c(0.25, 1, 2), ncol = 3, nrow =
             (matrix(c(0.25, 1, 2), ncol = 3, nrow = 100, byrow = TRUE) <= resultEM + (qnorm(1 - (0.05/
            colSums()) %>%
  t() %>%
  `rownames<-`(c("Standard Error", "Coverage Probability")) %>%
  kable(col.names = c("p", "$\\lambda$", "$\\mu$"),
        caption = "The standard error for each parameter by using the bootstrap Method")
```