Multiple Imputation Edge Cases

2022-07-22, Leo Murao Watson

Special Cases where Listwise Deletion is Preferred over Multiple Imputation

- Let Y = Ozone, X_1 = Wind, X_2 = Temp, X_3 = Month, X_4 = Day
- Will compare Missing Imputation and Listwise Deletion as missing data methods.
- Suppose scientifc interest focuses on determining β_1, β_2, β_3, β_4 in the linear model $y_i = \alpha + x_1\beta_1 + x_2\beta_2 + x_3\beta_3 + x_4\beta_4 + \epsilon_i$. *Here, ϵ_i $N(0, \sigma^2)$.

1) Exclusively Missing data in Response Y

```
# Draw data from artificial model specified above
create.data <- function(alpha = 20, beta_1 = 1, beta_2 = 2, beta_3 = 3, beta_4 = 4,
                        sigma2 = 1, n = 50, run = 1) {
  set.seed(seed = run)
  x_1 <- rnorm(n)
  x_2 <- rnorm(n)
  x_3 <- rnorm(n)
  x_4 <- rnorm(n)
  y <- beta_1 * x_1 + beta_2 * x_2 + beta_3 * x_3 + beta_4 * x_4 + alpha +
    rnorm(n, sd = sqrt(sigma2))
  cbind("Y" = y, "X_1"= x_1, "X_2" = x_2, "X_3" = x_3, "X_4" = x_4)
}

# Remove some data to simulate real world missingness
MCAR.make.missing <- function(data, p = 0.5){
  rx <- rbinom(nrow(data), 1, p)
  data[rx == 0, "Y"] <- NA
  data
}
```

Missing Imputation

```
# Simulate multiple imputation, obtaining estimates and 95% confidence interval.
simulate_MI2 <- function(runs = 100) {
  res <- array(NA, dim = c(5, runs, 3))
  times <- array(NA, dim = c(100, 1, 1))
  dimnames(res) <- list(c("Intercept", "X_1", "X_2", "X_3", "X_4"),
                        as.character(1:runs), c("estimate", "2.5%", "97.5%"))
  sim_dataset <- as.data.frame(create.data(n = 200))
  for (run in 1:runs){
      # Note that time is only measured for the MI/imp steps
      # (i.e. filtering, predicting)
    missingness_sim_dataset <- MCAR.make.missing(sim_dataset, p = 0.5)
    start_time <- Sys.time()
    imp_MI <- mice(missingness_sim_dataset, print = FALSE)
```

```r
    fit <- with(imp_MI, lm(Y ~ X_1 + X_2 + X_3 + X_4))
    end_time <- Sys.time()
    tab <- summary(pool(fit), "all", conf.int = TRUE)
    res[1, run, ] <- as.numeric(tab[1, c("estimate", "2.5 %", "97.5 %")])
    res[2, run, ] <- as.numeric(tab[2, c("estimate", "2.5 %", "97.5 %")])
    res[3, run, ] <- as.numeric(tab[3, c("estimate", "2.5 %", "97.5 %")])
    res[4, run, ] <- as.numeric(tab[4, c("estimate", "2.5 %", "97.5 %")])
    res[5, run, ] <- as.numeric(tab[5, c("estimate", "2.5 %", "97.5 %")])

    times[run, 1, 1] <- as.numeric(end_time - start_time)
  }
  list(res, times)
}
```

```r
# Run 100 iterations
res_MI2 <- simulate_MI2(100)
```

```r
# Obtain confidence intervals & estimates for all coefficients, intercept.
apply(res_MI2[[1]], c(1, 3), mean, na.rm = TRUE)
```

```
##              estimate       2.5%     97.5%
## Intercept 19.9953671 19.7273594 20.263375
## X_1        0.9366696  0.6472289  1.226110
## X_2        1.9149274  1.6500587  2.179796
## X_3        2.7973618  2.5456623  3.049061
## X_4        3.8961641  3.6515181  4.140810
```

```r
# Mean time for the multiple imputation instances
times <- res_MI2[[2]]
mean(times)
```

```
## [1] 0.07354221
```

```r
# Evaluating imputation method performance for estimating
# all parameters of interest.
res <- res_MI2[[1]]
true <- c(20, 1, 2, 3, 4)
RB <- rowMeans(res[,, "estimate"]) - true
PB <- 100 * abs((rowMeans(res[,, "estimate"]) - true)/ true)
CR <- rowMeans(res[,, "2.5%"] < true & true < res[,, "97.5%"])
AW <- rowMeans(res[,, "97.5%"] - res[,, "2.5%"])
RMSE <- sqrt(rowMeans((res[,, "estimate"] - true)^2))
data.frame(RB, PB, CR, AW, RMSE)
```

```
##                     RB         PB   CR        AW      RMSE
## Intercept -0.004632882 0.02316441 1.00 0.5360155 0.0955964
## X_1       -0.063330448 6.33304482 0.96 0.5788813 0.1180301
## X_2       -0.085072564 4.25362822 0.97 0.5297375 0.1353696
## X_3       -0.202638162 6.75460540 0.64 0.5033990 0.2373411
## X_4       -0.103835898 2.59589744 0.90 0.4892921 0.1475618
```

Listwise Deletion

```r
# Simulate listwise deletion, obtaining estimates and 95% confidence interval.

simulate_LD <- function(runs = 100){
```

```
  res <- array(NA, dim = c(5, runs, 3))
  dimnames(res) <- list(c("Intercept", "X_1", "X_2", "X_3", "X_4"),
                        as.character(1:runs), c("estimate", "2.5%", "97.5%"))
  times <- array(NA, dim = c(runs, 1, 1))
  sim_dataset <- as.data.frame(create.data(n = 200))
  # Note that time is only measured for the LD/imp steps (i.e. filtering, predicting)
  for (run in 1:runs){
    missingness_sim_dataset <- MCAR.make.missing(sim_dataset, p = 0.7)
    start_time <- Sys.time()
    filtered_sim_dataset <- missingness_sim_dataset %>%
      select(Y, X_1, X_2, X_3, X_4) %>%
      filter(!is.na(Y))
    fit <- with(filtered_sim_dataset, lm(Y ~ X_1 + X_2 + X_3 + X_4))
    end_time <- Sys.time()
    times[run, 1, 1] <- as.numeric(end_time - start_time)
    # loop over each variable. Note we do the imputation just ONCE b/c LD is
    # deterministic.
    for (var in 1:5){
      edges <- as.numeric((confint(fit)[var,]))
      estimate <- as.numeric(fit$coefficients)[var]
      interval <- c(estimate, edges)
      res[var, run, ] <- interval
      }

  }
  list(res, times)
}
```

```
result_LD <- simulate_LD()
```

```
# Obtain confidence intervals & estimates for all coefficients, intercept.
apply(result_LD[[1]], c(1, 3), mean, na.rm = TRUE)
```

```
##            estimate      2.5%     97.5%
## Intercept 20.014130 19.8316495 20.196610
## X_1        1.023355  0.8245039  1.222206
## X_2        2.069596  1.8887159  2.250476
## X_3        3.021985  2.8511103  3.192860
## X_4        4.033573  3.8658807  4.201265
```

```
# Evaluating imputation method performance for estimating
# all parameters of interest.
res <- result_LD[[1]]
true <- c(20, 1, 2, 3, 4)
RB <- rowMeans(res[,, "estimate"]) - true
PB <- 100 * abs((rowMeans(res[,, "estimate"]) - true)/ true)
CR <- rowMeans(res[,, "2.5%"] < true & true < res[,, "97.5%"])
AW <- rowMeans(res[,, "97.5%"] - res[,, "2.5%"])
RMSE <- sqrt(rowMeans((res[,, "estimate"] - true)^2))
data.frame(RB, PB, CR, AW, RMSE)
```

```
##                   RB          PB   CR        AW       RMSE
## Intercept 0.01412967 0.07064837 1.00 0.3649604 0.04946551
## X_1       0.02335495 2.33549497 1.00 0.3977020 0.05633452
## X_2       0.06959586 3.47979321 0.98 0.3617599 0.08869656
```

```
## X_3       0.02198526 0.73284197 1.00 0.3417499 0.04601087
## X_4       0.03357291 0.83932276 1.00 0.3353844 0.05205810
```

```r
# Mean time for 100 instances of LD
times_LD <- result_LD[[2]]
mean(times_LD)
```

```
## [1] 0.007305567
```