assn2

2023-11-19

Question1

Data preparation

```
# Data preparation
library(rstan)
```

```
## Loading required package: StanHeaders
```

```
##
## rstan version 2.32.3 (Stan version 2.26.1)
```

```
## For execution on a local, multicore CPU with excess RAM we recommend calling
## options(mc.cores = parallel::detectCores()).
## To avoid recompilation of unchanged Stan programs, we recommend calling
## rstan_options(auto_write = TRUE)
## For within-chain threading using ‘reduce_sum()‘ or ‘map_rect()‘ Stan functions,
## change ‘threads_per_chain‘ option:
## rstan_options(threads_per_chain = 1)
```

```
options(mc.cores = parallel::detectCores())
rstan_options(auto_write = TRUE)
library(tidyverse)
```

```
## -- Attaching packages --------------------------------------- tidyverse 1.3.2 --
```

```
## v ggplot2 3.4.0     v purrr   0.3.4
## v tibble  3.2.1     v dplyr   1.1.2
## v tidyr   1.2.1     v stringr 1.4.1
## v readr   2.1.2     v forcats 0.5.2
## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x tidyr::extract() masks rstan::extract()
## x dplyr::filter()  masks stats::filter()
## x dplyr::lag()     masks stats::lag()
```

```
library(mvtnorm)
data <- read_csv("score.csv")
```

```
## Rows: 2287 Columns: 5
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## dbl (5): Class, IQ, SCORE, SES, Gender
##
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

```
data <- data %>%
  left_join(data %>%
              group_by(Class) %>%
              summarise(IQCL = mean(IQ)) %>%
              ungroup(),
            by = "Class") %>%
  mutate(Class = as.integer(Class),
         Gender = as.integer(Gender)); data
```

```
## # A tibble: 2,287 x 6
##    Class    IQ SCORE   SES Gender  IQCL
##    <int> <dbl> <dbl> <dbl>  <int> <dbl>
##  1     1 14.3   46.5    11      0  11.3
##  2     1 14.1   50.7     0      1  11.3
##  3     1  4.11  28.6    10      1  11.3
##  4     1 15.4   42.7     9      1  11.3
##  5     1  4.09  35.5    10      1  11.3
##  6     1 17.5   44.9     3      0  11.3
##  7     1  6.04  37.6     9      1  11.3
##  8     1  7.29  26.3     6      1  11.3
##  9     1 16.9   50.4     5      1  11.3
## 10     1 13.7   45.0     6      1  11.3
## # i 2,277 more rows
```

```
list_data <- list(N = nrow(data),
                  L = nrow(distinct(data, Class)),
                  ll = data$Class,
                  score = data$SCORE,
                  gender = data$Gender,
                  iq = data$IQ,
                  ses = data$SES,
                  iqcl = distinct(data, IQCL)$IQCL,
                  D1 = 2,
                  D2 = 3,
                  D3 = 2)
```

Model(a) with random effects:

$$
\begin{cases}
y_{ij} \sim \mathcal{N}\left(\mu_{ij}, V_{ij}\right), \quad i = 1, \ldots, n_j, j = 1, \ldots, J \\
\mu_{ij} = b_{j1} + b_{j2}\left(\mathrm{IQ}_{ij} - \overline{\mathrm{IQ}}\right) + \beta_1\left(\mathrm{SES}_{ij} - \overline{\mathrm{SES}}\right) + \beta_2 \mathrm{G}_{ij} + \beta_3 \mathrm{IQCL}_j \\
\left(b_{j1}, b_{j2}\right) \sim \mathcal{N}_2\left(\left[m_1, m_2\right], \Sigma_b\right) \\
V_{ij} = \theta_1 + \theta_2 \mathrm{IQ}_{ij}
\end{cases}
$$

```
fit.m1 <- stan(file = "./q1m1.stan",data = list_data)
```

```
## Warning in readLines(file, warn = TRUE): incomplete final line found on '/Users/
## jiqi/Library/Mobile Documents/com~apple~CloudDocs/Course/STAT5060/q1m1.stan'
```

```
## Warning: There were 540 divergent transitions after warmup. See
## https://mc-stan.org/misc/warnings.html#divergent-transitions-after-warmup
## to find out why this is a problem and how to eliminate them.
```

```
## Warning: There were 10 transitions after warmup that exceeded the maximum treedepth. Increase max_tr
## https://mc-stan.org/misc/warnings.html#maximum-treedepth-exceeded
```

```
## Warning: There were 4 chains where the estimated Bayesian Fraction of Missing Information was low. S
## https://mc-stan.org/misc/warnings.html#bfmi-low
```

```
## Warning: Examine the pairs() plot to diagnose sampling problems
```

```
## Warning: The largest R-hat is 1.32, indicating chains have not mixed.
## Running the chains for more iterations may help. See
## https://mc-stan.org/misc/warnings.html#r-hat
```

```
## Warning: Bulk Effective Samples Size (ESS) is too low, indicating posterior means and medians may be
## Running the chains for more iterations may help. See
## https://mc-stan.org/misc/warnings.html#bulk-ess
```

```
## Warning: Tail Effective Samples Size (ESS) is too low, indicating posterior variances and tail quant
## Running the chains for more iterations may help. See
## https://mc-stan.org/misc/warnings.html#tail-ess
```

Model(b) without random effects:

$$\begin{cases} y_{ij} \sim \mathcal{N}\left(\mu_{ij}, V_{ij}\right), \quad i = 1, \ldots, n_j, j = 1, \ldots, J \\ \mu_{ij} = m_1 + m_2\left(\mathrm{IQ}_{ij} - \overline{\mathrm{IQ}}\right) + \beta_1\left(\mathrm{SES}_{ij} - \overline{\mathrm{SES}}\right) + \beta_2 \mathrm{G}_{ij} + \beta_3 \mathrm{IQCL}_j \\ V_{ij} = \theta_1 + \theta_2 \mathrm{IQ}_{ij} \end{cases}$$

```
fit.m2 <- stan(file = "./q1m2.stan",data = list_data)
```

```
## Warning in readLines(file, warn = TRUE): incomplete final line found on '/Users/
## jiqi/Library/Mobile Documents/com~apple~CloudDocs/Course/STAT5060/q1m2.stan'
```

Comparison of the two models

```
summary(fit.m1)$summary[1:11, c("mean", "sd", "2.5%", "97.5%")]
```

```
##                       mean         sd         2.5%        97.5%
## m[1]          3.719402e+01 1.65474356 33.988905324  40.37971146
## m[2]          1.995851e+00 0.04098236  1.921092804   2.07468873
## sigma_b[1,1]  4.669333e-01 0.31143975  0.103028169   1.23368002
## sigma_b[1,2] -1.021298e-02 0.05654676 -0.147699731   0.08034476
## sigma_b[2,1] -1.021298e-02 0.05654676 -0.147699731   0.08034476
## sigma_b[2,2]  2.412687e-02 0.01915108  0.002219272   0.07156074
## beta[1]       3.570464e-05 0.04239283 -0.087117904   0.07978411
## beta[2]       2.131308e-01 0.28634468 -0.347469806   0.78246160
## beta[3]       1.402753e-01 0.15102446 -0.156406678   0.43532113
## theta[1]      1.047743e+02 5.33035470 94.428614923 115.32639013
## theta[2]     -4.635212e+00 0.37331288 -5.351072402  -3.91833812
```

```
summary(fit.m2)$summary[1:7, c("mean", "sd", "2.5%", "97.5%")]
```

```
##                    mean         sd        2.5%        97.5%
## m[1]      37.342743534 1.69374351 33.97433093  40.68541928
## m[2]       1.986115362 0.03683460  1.91326071   2.05856411
## beta[1]   -0.001596913 0.04123526 -0.08136143   0.08162905
## beta[2]    0.222076493 0.29082761 -0.34400435   0.77734807
## beta[3]    0.120657272 0.15317207 -0.17675011   0.42502320
## theta[1] 104.667726223 5.38902150 94.10583900 115.32927359
## theta[2]  -4.586092841 0.37614370 -5.29223384  -3.83857350
```

Similarities:

- The 95% CI of m_1 does not cover 0 in both models, which suggests that centered IQ is a significant variable when predicting y_{ij}.
- The 95% CI of $\beta_1, \beta_2, \beta_3$ cover 0 in both models, which suggests that SES, gender, and average IQ of class are not significant.
- The 95% CI of θ_2 does not cover 0 in both models and θ_2 are both negative, which suggests that as IQ increases, the variance of y_ij are relatively decreases.

Differences:

- Σ_{11} and Σ_{22} are both significant, which suggests the randomness of intercept and slope is necessary for fitting the model.
- Σ_{12} is not significant in Model(a), which suggests we can assume that Σ is diagonal to get better fitted model.

Question2

```
library(mvtnorm)

# Set parameters and specifications
n <- 800
pi1 <- 0.5
```

```r
pi2 <- 0.5
sigma1 <- diag(0.2, 3)
sigma2 <- diag(0.2, 3)
alpha1 <- matrix(c(1, -2, 1), nrow = 3)
alpha2 <- matrix(c(-2, 1, 2), nrow = 3)
beta1 <- matrix(c(-1, 1, 1, -1, 1, 1), nrow = 3)
beta2 <- matrix(c(1, 2, 3, 1, 2, 3), nrow = 3)

# Generate independent variables
xi1 <- rnorm(n)
xi2 <- runif(n)

# Generate mixture component indicators
Si <- sample(c(1, 2), n, replace = TRUE, prob = c(pi1, pi2))

# Generate response variables
yi <- matrix(0, nrow = n, ncol = 3)
for (i in 1:n) {
  if (Si[i] == 1) {
    epsilon <- matrix(c(rmvnorm(1, mean = rep(0, 3), sigma = sigma1)), nrow = 3)
    yi[i, ] <- alpha1 + beta1 %*% c(xi1[i], xi2[i]) + epsilon
  } else {
    epsilon <- matrix(c(rmvnorm(1, mean = rep(0, 3), sigma = sigma2)), nrow = 3)
    yi[i, ] <- alpha2 + beta2 %*% c(xi1[i], xi2[i]) + epsilon
  }
}

# View the simulated dataset
simulated_data <- data.frame(yi)
head(simulated_data)
```

```
##            X1         X2        X3
## 1  0.48866456 -2.1332722 0.8852028
## 2  0.44131839 -0.6608000 1.1172782
## 3 -1.72000524  1.5067344 3.2762576
## 4 -0.15844429 -1.4772724 2.4087850
## 5 -1.01624280  3.3448778 3.9265553
## 6  0.09253693 -0.9658718 0.7278017
```

Since Σ_k is diagonal, we can run mixEM algorithm by row:

```r
library(mixtools)
```

```
## mixtools package, version 2.0.0, Released 2022-12-04
## This package is based upon work supported by the National Science Foundation under Grant No. SES-0518

##
## Attaching package: 'mixtools'

## The following objects are masked from 'package:mvtnorm':
##
##     dmvnorm, rmvnorm
```

```
xi <- matrix(c(xi1, xi2), ncol = 2)

# Estimate the mixture model parameters
fitlist <- list()
for (i in 1:3) {
  fit <- regmixEM(yi[, i], xi, lambda = NULL, beta = NULL, sigma = NULL, k = 2,
                  addintercept = TRUE, arbmean = TRUE, arbvar = TRUE,
                  epsilon = 1e-08, maxit = 10000, verb = FALSE)
  fitlist[[i]] <- fit
}
```

```
## number of iterations= 12
## number of iterations= 12
## number of iterations= 15
```

```
alpha1_est <- matrix(ncol = 1, nrow = 3)
alpha2_est <- matrix(ncol = 1, nrow = 3)
beta1_est <- matrix(ncol = 2, nrow = 3)
beta2_est <- matrix(ncol = 2, nrow = 3)
pi_est = matrix(ncol = 2, nrow = 3)
sigma_est = matrix(ncol = 2, nrow = 3)
for (i in 1:3){
  pi_est[i, ] = fitlist[[i]]$lambda
  alpha1_est[i, 1] = fitlist[[i]]$beta[1, 1]
  alpha2_est[i, 1] = fitlist[[i]]$beta[1, 2]
  beta1_est[i, ] = fitlist[[i]]$beta[2:3, 1]
  beta2_est[i, ] = fitlist[[i]]$beta[2:3, 2]
  sigma_est[i, 1] = fitlist[[i]]$lambda[1] ^ 2
  sigma_est[i, 2] = fitlist[[i]]$lambda[2] ^ 2
}

pi_est = colMeans(pi_est)
# Print caption and the result of pi_est
cat("Estimated pi: ")
```

```
## Estimated pi:
```

```
print(pi_est)
```

```
## [1] 0.5072449 0.4927551
```

```
# Print caption and the result of alpha1_est
cat("Estimated alpha1: ")
```

```
## Estimated alpha1:
```

```
print(alpha1_est)
```

```
##           [,1]
## [1,] -2.024850
## [2,] -1.908488
## [3,]  1.030613
```

```r
# Print caption and the result of alpha2_est
cat("Estimated alpha2: ")
```

```
## Estimated alpha2:
```

```r
print(alpha2_est)
```

```
##           [,1]
## [1,] 1.0416540
## [2,] 0.9786971
## [3,] 2.0210465
```

```r
# Print caption and the result of beta1_est
cat("Estimated beta1: ")
```

```
## Estimated beta1:
```

```r
print(beta1_est)
```

```
##           [,1]      [,2]
## [1,] 0.9503859 1.0206757
## [2,] 0.9911187 0.8002596
## [3,] 1.0144404 0.9171200
```

```r
# Print caption and the result of beta2_est
cat("Estimated beta2: ")
```

```
## Estimated beta2:
```

```r
print(beta2_est)
```

```
##            [,1]      [,2]
## [1,] -0.9753518 -1.119752
## [2,]  2.0015382  1.981046
## [3,]  2.9359089  3.001556
```

Since there is no constraint on α_k and β_k and Σ_1, Σ_2 are both diagonal, the estimations of α_k and β_k are not identifiable for each rows. If we switch the first row of α_1 and α_2, β_1 and β_2, our estimations are very close to the real parameters. One way is to add a prior or constraint(e.g. $\alpha_{k1} > \alpha_{k2}$) to avoid unidentifiability.

Question3

Data preparation

```r
library(tidyverse)
library(rstan)
L <- 2
m <- 400
N <- L * m
D1 <- 2
mu <- 0
sd_main <- 0.5
beta <- c(1, -1)
u <- c(1, -1)
x <- matrix(rnorm(n = N * D1), ncol = D1)
z <- matrix(runif(n = N))
y <- mu + x %*% beta + z * rep(u, each = m) + rnorm(N, mean = 0, sd = sd_main) ## missing generating
c <- -1
alpha <- -1
gamma <- c(1, -1)
R <- boot::inv.logit(c + alpha * y + x %*% gamma) %>% sapply(function (x) rbinom(n = 1, size = 1, prob =

list_data <- list(N = N,
                  N_obs = sum(R == 0),
                  N_mis = sum(R == 1),
                  L = L,
                  D1 = D1,
                  y_obs = y[R == 0],
                  x = x,
                  z = as.vector(z),
                  ll = rep(1:D1, each = m),
                  R = R,
                  ii_obs = which(R == 0),
                  ii_mis = which(R == 1))
fit.mnar <- stan(file = "q3MNAR.stan", data = list_data)
```

```
## Warning in readLines(file, warn = TRUE): incomplete final line found on '/Users/
## jiqi/Library/Mobile Documents/com~apple~CloudDocs/Course/STAT5060/q3MNAR.stan'
```

```r
fit.mar <- stan(file = "q3MAR.stan",data = list_data)
```

```
## Warning in readLines(file, warn = TRUE): incomplete final line found on '/Users/
## jiqi/Library/Mobile Documents/com~apple~CloudDocs/Course/STAT5060/q3MAR.stan'

## hash mismatch so recompiling; make sure Stan code ends with a blank line
```

```r
mean(R)
```

```
## [1] 0.3025
```

The missing proportion is approximately 30%.

Model(a) MNAR

```
summary(fit.mnar)$summary[1:10, c("mean", "sd", "2.5%", "97.5%")]
```

```
##                mean         sd        2.5%      97.5%
## mu         0.02179837 0.04260659 -0.06312106  0.1062095
## beta[1]    1.01959405 0.02069745  0.97857029  1.0590973
## beta[2]   -0.99231558 0.02109047 -1.03441290 -0.9515455
## sigma      0.50518454 0.01632040  0.47446162  0.5382116
## u[1]       1.00459435 0.07868407  0.84800357  1.1591734
## u[2]      -0.97832577 0.09197549 -1.16383179 -0.7966476
## c         -0.92195336 0.09372134 -1.11323618 -0.7401036
## alpha     -1.06599061 0.16839070 -1.40552062 -0.7423006
## gamma[1]   1.19289538 0.19288900  0.82207458  1.5850786
## gamma[2]  -1.08100186 0.18403201 -1.45220212 -0.7251376
```

Model(b) MAR

```
summary(fit.mar)$summary[1:9, c("mean", "sd", "2.5%", "97.5%")]
```

```
##                mean         sd          2.5%      97.5%
## mu         0.08390150 0.04084731 -0.0002717985  0.1615210
## beta[1]    1.02464003 0.02108104  0.9846102926  1.0652663
## beta[2]   -0.99410576 0.02047226 -1.0346828163 -0.9533784
## sigma      0.49225199 0.01507825  0.4640064493  0.5229795
## u[1]       0.98871156 0.07845210  0.8350974867  1.1437485
## u[2]      -0.89087413 0.09025699 -1.0611691229 -0.7115025
## c         -0.84368796 0.07715731 -0.9978384154 -0.6934705
## gamma[1]   0.11375564 0.07800568 -0.0396564239  0.2675612
## gamma[2]  -0.02587847 0.07481529 -0.1744396438  0.1214102
```

Comparison

- The 95% CI of γ_1, γ_2 and c in MNAR model covers the true parameters, while in MAR model the 95% CI does not cover the true parameter. It means MNAR model can more precisely estimate the true parameters.
- Compared to MAR model, MNAR model consider the effect of covariates X on the missing mechanism and estimate the coefficient α when modeling

$$\text{logit}\left(\pi_{ij}\right) = c + \alpha y_{ij} + \gamma^T \mathbf{x}_{ij}, \quad \pi_{ij} = p\left(R_{ij} = 1\right),$$

  which better calculates $P(R|Y, X, \eta)$ and improves the accuracy of the estimation.