Handling Missing Data in Health Science Research

Day 2 - Part II

2022-06-23

Contents

MI for longitudinal data

- In the IPW example, all the covariates were fully observed
- However, what if some of the covariates were also missing?
- MI can handle situations where the longitudinal outcomes and subject-level covariates are both missing
- We will use `skinl_MAR.txt` where we simulated `age` and `skin` to have missing values

```
skin_long <- read.table("skinl_mar.txt", header = TRUE)
skin_long[, c("center","skin","gender","treatment","Y")] <-
  lapply(skin_long[, c("center","skin","gender","treatment","Y")], factor)
head(skin_long)
```

```
##       ID center age skin gender exposure treatment year Y
## 1 100034      1  NA    1      1        4         0    1 0
## 2 100034      1  NA    1      1        4         0    2 1
## 3 100034      1  NA    1      1        4         0    3 1
## 4 100034      1  NA    1      1        4         0    4 1
## 5 100034      1  NA    1      1        4         0    5 0
## 6 100045      1  68    1      0        2         0    1 0
```

- To make the example a little more applicable (and simpler), we will consider a different analysis model for the MI method

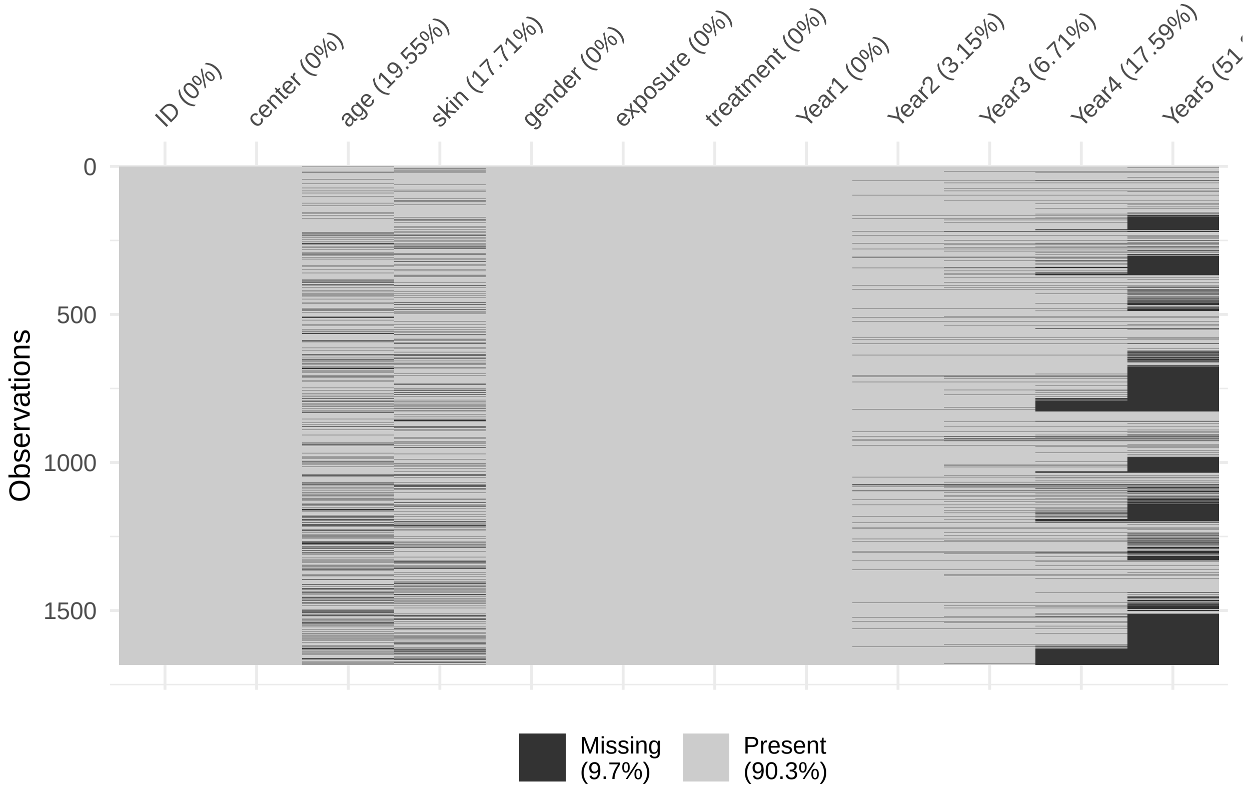
  $$\text{logit}(\mu_{ij}) = \beta_0 + \beta_1 \text{age}_i + \beta_2 \text{skin}_i + \beta_3 \text{exposure}_i + \beta_4 \text{treatment}_i + \beta_5 \text{Year}_{ij} + \beta_5 \text{Year}_{ij}^2, \quad j = 2, 3, 4, 5$$

  where $\mu_{ij} = \Pr(Y_{ij} = 1)$ is the probability of developing skin cancer for the ith subject in jth year

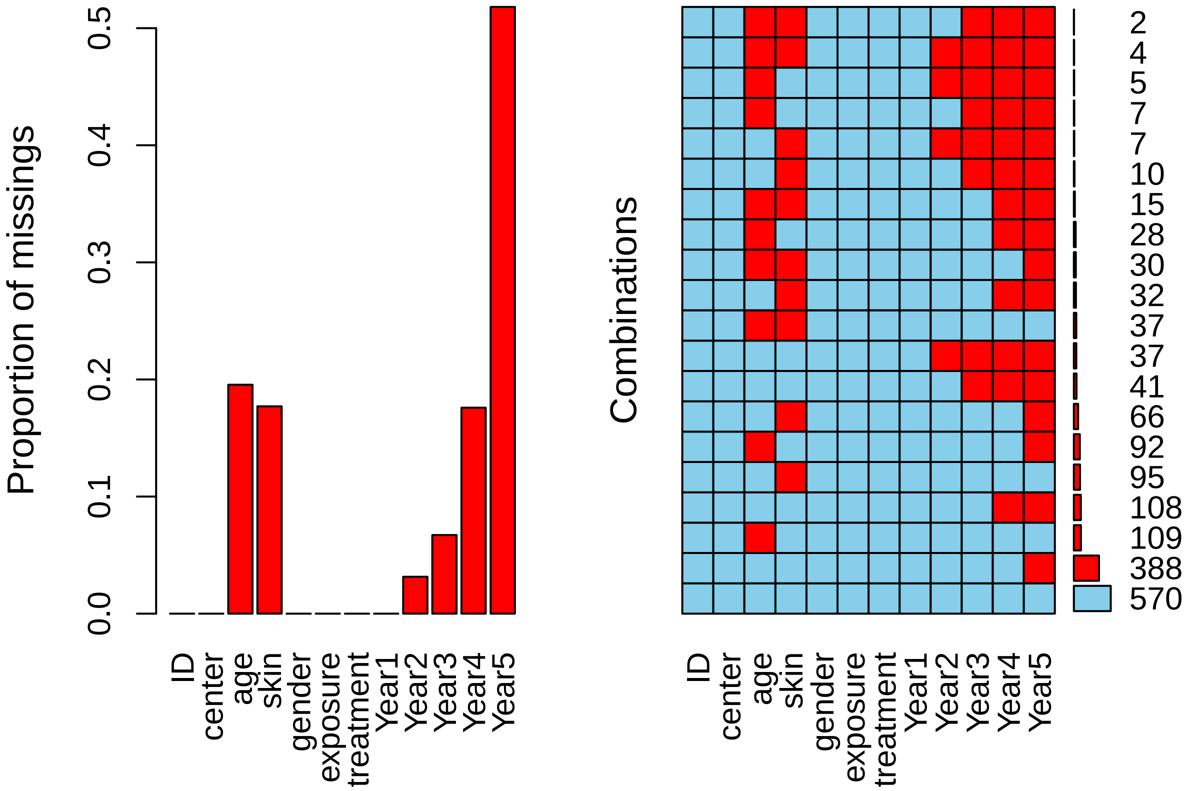
- Note that in addition to the outcomes, some of the predictors (age, skin) are also missing
- Mice and jomo are able to handle MI for multilevel data
  - Level 1 variables do not repeat within a subject (cluster)
  - Y is a “level 1” variable
  - Level 2 variables repeat within a subject (cluster)
  - Age and skin are “level 2” variables

Visualization

```
# need data to be in wide format
skin_wide <- skin_long %>%
  pivot_wider(names_from = year,
              names_prefix = "Year",
              values_from = Y)
# heatmap
naniar::vis_miss(skin_wide)
```



```
# aggregation plot
VIM::aggr(skin_wide, numbers = TRUE, prop = c(TRUE, FALSE))
```



Multilevel imputation

Mice

- We will first try imputation without treating the data as multilevel data
- We will use logistic regression to predict the binary outcome Y

```
skin_long$year2 <- skin_long$year**2
head(skin_long)
```

```
##       ID center age skin gender exposure treatment year Y year2
## 1 100034      1  NA    1      1        4         0    1 0     1
## 2 100034      1  NA    1      1        4         0    2 1     4
## 3 100034      1  NA    1      1        4         0    3 1     9
## 4 100034      1  NA    1      1        4         0    4 1    16
## 5 100034      1  NA    1      1        4         0    5 0    25
## 6 100045      1  68    1      0        2         0    1 0     1
```

```
start.time <- Sys.time()
mice.multi.out1 <- mice(skin_long, seed = 1, m = 5,
                        method = c("", "", "norm", "logreg", "", "", "", "", "logreg", ""),
                        print = F)
end.time <- Sys.time()
end.time - start.time
```

```
## Time difference of 9.646803 secs
```

```
complete.data.multi1 <- complete(mice.multi.out1, "all")

## check the first imputed dataset
head(complete.data.multi1$`1`, 20)
```

```
##        ID center      age skin gender exposure treatment year Y year2
## 1  100034      1 60.88025    1      1        4         0    1 0     1
## 2  100034      1 69.14562    1      1        4         0    2 1     4
## 3  100034      1 58.08112    1      1        4         0    3 1     9
## 4  100034      1 67.06986    1      1        4         0    4 1    16
## 5  100034      1 75.24804    1      1        4         0    5 0    25
## 6  100045      1 68.00000    1      0        2         0    1 0     1
## 7  100045      1 68.00000    1      0        2         0    2 0     4
## 8  100045      1 68.00000    1      0        2         0    3 0     9
## 9  100045      1 68.00000    1      0        2         0    4 0    16
## 10 100045      1 68.00000    1      0        2         0    5 0    25
## 11 100056      1 58.00000    1      0        7         0    1 1     1
## 12 100056      1 58.00000    1      0        7         0    2 1     4
## 13 100056      1 58.00000    1      0        7         0    3 0     9
## 14 100056      1 58.00000    1      0        7         0    4 1    16
## 15 100056      1 58.00000    1      0        7         0    5 0    25
## 16 100067      1 53.00000    1      1        3         0    1 0     1
## 17 100067      1 53.00000    1      1        3         0    2 0     4
## 18 100067      1 53.00000    1      1        3         0    3 0     9
## 19 100067      1 53.00000    1      1        3         0    4 0    16
## 20 100067      1 53.00000    1      1        3         0    5 0    25
```

- Note that when you treat all observations as independent, the imputed values for `age` for ID = 100034 are all different at each year

```
## geeglm() requires numeric (0/1) rather than factor for binary data
## we first use as.numeric(as.character()) to transfer data into 0/1
fit.multi1 <- with(mice.multi.out1,
                   model1 <- geeglm(as.numeric(as.character(Y)) ~ age + skin +
                                      treatment + gender + exposure + year + year2,
                                    id = ID, scale.fix = TRUE,
                                    corstr = "ar1",
                                    family = binomial("logit")))

testEstimates(as.mitml.result(fit.multi1))
```

```
##
## Call:
##
## testEstimates(model = as.mitml.result(fit.multi1))
##
## Final parameter estimates and inferences obtained from 5 imputed data sets.
##
##              Estimate Std.Error   t.value        df   P(>|t|)       RIV       FMI
## (Intercept)    -3.237     0.391    -8.269    17.182     0.000     0.932     0.534
## age             0.011     0.005     2.271    17.615     0.036     0.910     0.527
## skin1           0.192     0.125     1.537     8.666     0.160     2.119     0.734
## treatment1      0.089     0.079     1.124   290.231     0.262     0.133     0.123
## gender1         0.622     0.098     6.374   133.633     0.000     0.209     0.185
## exposure        0.164     0.012    13.868   205.849     0.000     0.162     0.148
## year           -0.223     0.118    -1.891    70.390     0.063     0.313     0.259
## year2           0.041     0.021     1.968    39.263     0.056     0.469     0.351
##
## Unadjusted hypothesis test as appropriate in larger samples.
```

- Now we will perform MI by treating the data as multilevel data
- We still use logistic regression for outcome Y
- A useful feature of the `mice()` function is the ability to specify the set of predictors to be used for each incomplete variable
- The basic specification is made through the `predictorMatrix` argument, which is a square matrix of size `ncol(data)` containing 0/1 data
- Each row in `predictorMatrix` identifies which predictors are to be used for the variable in the row name.

```
pred <- mice.multi.out1$predictorMatrix
pred
```

```
##           ID center age skin gender exposure treatment year Y year2
## ID         0      1   1    1      1        1         1    1 1     1
## center     1      0   1    1      1        1         1    1 1     1
## age        1      1   0    1      1        1         1    1 1     1
## skin       1      1   1    0      1        1         1    1 1     1
## gender     1      1   1    1      0        1         1    1 1     1
## exposure   1      1   1    1      1        0         1    1 1     1
## treatment  1      1   1    1      1        1         0    1 1     1
## year       1      1   1    1      1        1         1    0 1     1
## Y          1      1   1    1      1        1         1    1 0     1
## year2      1      1   1    1      1        1         1    1 1     0
```

- For example, age is predicted from center, skin, gender, exposure, treatment, year and Y
- gender does not have missing value, so gender will not be imputed
- For two-level imputation models, other numeric codes are allowed

Allowed entries in predictorMatrix:

Setting	Description
-2	class variable (only one is allowed, must be indicated when you perform multilevel imputation)
0	variable is not included in the imputation model
1	variable is included as a fixed effect
2	variable is included as a random effect
3	variable is included as a fixed effect and group mean
4	variable is included as a random effect and group mean

The above information is obtained from help(mice.impute.2l.pan) and https://bookdown.org/mwheymans/bookmi/multiple-imputation-models-for-multilevel-data.html#the-predictormatrix

```r
## change the code to specify our own imputation model
## you have to specify class variables when using multilevel imputation
pred[c("age", "skin", "Y"), "ID"] <- -2
pred
```

```
##           ID center age skin gender exposure treatment year Y year2
## ID         0      1   1    1      1        1         1    1 1     1
## center     1      0   1    1      1        1         1    1 1     1
## age       -2      1   0    1      1        1         1    1 1     1
## skin      -2      1   1    0      1        1         1    1 1     1
## gender     1      1   1    1      0        1         1    1 1     1
## exposure   1      1   1    1      1        0         1    1 1     1
## treatment  1      1   1    1      1        1         0    1 1     1
## year       1      1   1    1      1        1         1    0 1     1
## Y         -2      1   1    1      1        1         1    1 0     1
## year2      1      1   1    1      1        1         1    1 1     0
```

```r
## convert the factor of skin into numeric to avoid NA when using PMM
## 2lonly is for imputing level-2 variable, 2l is for imputing level-1 variable
skin_long2 <- skin_long %>%
  mutate(skin = as.numeric(as.character(skin)))
start.time <- Sys.time()
mice.out  <- mice(skin_long2, seed = 100, m = 5,
                  method = c("", "", "2lonly.norm", "2lonly.pmm","","","","","2l.bin", ""),
                  predictorMatrix = pred, print = F, maxit = 5)
end.time <- Sys.time()
end.time - start.time
```

```
## Time difference of 57.11626 mins
```

```r
## The warning results from the linear dependencies among the predictors.
## The mice() function checks for linear dependencies during the iterations,
## and temporarily removes predictors from the univariate imputation models where needed.

complete.data.multi2 <- complete(mice.out, "all")

##check the first imputed dataset
head(complete.data.multi2$`1`)
```

```
##       ID center     age skin gender exposure treatment year Y year2
## 1 100034      1 87.2498    1      1        4         0    1 0     1
## 2 100034      1 87.2498    1      1        4         0    2 1     4
## 3 100034      1 87.2498    1      1        4         0    3 1     9
## 4 100034      1 87.2498    1      1        4         0    4 1    16
## 5 100034      1 87.2498    1      1        4         0    5 0    25
## 6 100045      1 68.0000    1      0        2         0    1 0     1
```

```r
## fit analysis model and pool results
fit.multi2 <- with(mice.out,
                   model1 <- geeglm(as.numeric(as.character(Y)) ~ age + skin +
                                      treatment + gender + exposure + year + year2,
                                    id = ID, scale.fix = TRUE,
                                    corstr = "ar1",
                                    family = binomial("logit")))
testEstimates(as.mitml.result(fit.multi2))
```

```
##
## Call:
##
## testEstimates(model = as.mitml.result(fit.multi2))
##
## Final parameter estimates and inferences obtained from 5 imputed data sets.
##
##              Estimate Std.Error   t.value        df   P(>|t|)       RIV       FMI
## (Intercept)    -2.635     1.503    -1.753     4.223     0.151    36.303     0.981
## age            -0.000     0.024    -0.002     4.116     0.998    69.726     0.990
## skin            0.169     0.114     1.486    13.333     0.161     1.211     0.603
## treatment1      0.054     0.082     0.658   205.548     0.511     0.162     0.148
## gender1         0.595     0.095     6.256   534.247     0.000     0.095     0.090
## exposure        0.167     0.013    13.071    74.235     0.000     0.302     0.252
## year           -0.167     0.121    -1.384    71.717     0.171     0.309     0.257
## year2           0.026     0.021     1.239    39.383     0.223     0.468     0.351
##
## Unadjusted hypothesis test as appropriate in larger samples.
```

A note about FCS mulilevel imputation

MICE provides limited choice for imputing multilevel data and it could also be very time consuming when iteration times is large. Recently a new software called Blimp has been developed for imputing multilevel data using FCS (https://pubmed.ncbi.nlm.nih.gov/28557466/). It provides different options for imputing continuous variables, categorical variables (including nominal and ordinal).

Joint Modeling

- We will now use the R package jomo to perform joint modeling (JM)
- This package can impute multilevel data
- When imputing the multilevel data using jomo, we need to distinguish between Y and Y2, X and X2
  - Y is for level-1 missing data
  - Y2 is for level-2 missing data
  - X is for level-1 complete data
  - X2 is for level-2 complete data

```
## be careful about the column names of each dataframe
## jomo will inherit from those column names
Y.jomo.l1miss <- data.frame(Y = skin_long[,c("Y")])
Y.jomo.l2miss <- data.frame(skin_long[,c("age", "skin")])

X.jomo.l1complete <- data.frame(skin_long[,c("year", "year2")])
X.jomo.l2complete <- data.frame(skin_long[,c("gender", "exposure", "treatment")])
## removing this will change the result
X.jomo.l2complete$const <- 1

set.seed(100)
start.time <- Sys.time()
```

```
jomo.multi.out <- jomo(Y = Y.jomo.l1miss, Y2 = Y.jomo.l2miss,
                       X = X.jomo.l1complete, X2 = X.jomo.l2complete,
                       clus = skin_long$ID, nburn = 1000, nbetween = 1000, nimp = 5)
end.time <- Sys.time()
end.time - start.time
```

```
head(jomo.multi.out[which(jomo.multi.out$Imputation == 1),])
```

```
##      Y      age skin year year2 gender exposure treatment const Z1   clus id
## 8416 0 60.31204    1    1     1      2        4         1     1  1 100034  1
## 8417 1 60.31204    1    2     4      2        4         1     1  1 100034  2
## 8418 1 60.31204    1    3     9      2        4         1     1  1 100034  3
## 8419 1 60.31204    1    4    16      2        4         1     1  1 100034  4
## 8420 0 60.31204    1    5    25      2        4         1     1  1 100034  5
## 8421 0 68.00000    1    1     1      1        2         1     1  1 100045  6
##      Imputation
## 8416          1
## 8417          1
## 8418          1
## 8419          1
## 8420          1
## 8421          1
```

```
## use imputationList from mitools() to split the imputed dataset
imp.list.jomo.multi <- imputationList(split(jomo.multi.out, jomo.multi.out$Imputation)[-1])

## fit the analysis model
fit.jomo.multi <- with(imp.list.jomo.multi,
                       model1 <- geeglm(as.numeric(as.character(Y)) ~ age + skin +
                                          treatment + gender + exposure + year + year2,
                                        id = clus, scale.fix = TRUE,
                                        corstr = "ar1",
                                        family = binomial("logit")))

## pool the results
testEstimates(as.mitml.result(fit.jomo.multi))
```

```
## 
## Call:
## 
## testEstimates(model = as.mitml.result(fit.jomo.multi))
## 
## Final parameter estimates and inferences obtained from 5 imputed data sets.
## 
##                Estimate Std.Error   t.value        df   P(>|t|)       RIV       FMI
## (Intercept)    -3.724     0.403    -9.245    71.924     0.000     0.309     0.256
## age             0.013     0.005     2.919    60.500     0.005     0.346     0.281
## skin1           0.224     0.101     2.221    24.214     0.036     0.685     0.450
## treatment       0.048     0.080     0.601   418.025     0.548     0.108     0.102
## gender          0.569     0.097     5.899   145.461     0.000     0.199     0.177
## exposure        0.139     0.012    11.888   181.749     0.000     0.174     0.158
## year           -0.378     0.102    -3.697  3779.461     0.000     0.034     0.033
## year2            0.072     0.017     4.256  1209.779     0.000     0.061     0.059
##
## Unadjusted hypothesis test as appropriate in larger samples.
```

A note about imputing Poisson data The implementation of Poisson data is not well developed in the joint modeling framework. However, count variables can be treated as categorical variables or continuous variables, suggested by Quartagno et al. (Multiple imputation for discrete data: Evaluation of the joint latent normal model). The former is only viable when the mean of the underlying Poisson distribution is low. It is generally thought that Poisson distributions with mean > 20 can be well approximated by the normal distribution. In this case variance-stabilizing square-root transformation or log transformation can be used.

Methods for data MNAR

- Identify additional data to make the data more "MAR"
- Perform sensitivity analysis: explore the results of the analysis under alternative scenarios for missing data (simulation studies)
- If the above has been investigated, we can use nonignorable imputation model, which models the distribution $\Pr(Y, R)$ instead of $P(Y)$ (R is the missing indicator)

  1. selection model: $\Pr(Y, R) = \Pr(Y)\Pr(R|Y)$
  2. pattern-mixture model: $\Pr(Y, R) = \Pr(Y|R)\Pr(R) = \Pr(Y|R = 1)\Pr(R = 1) + \Pr(Y|R = 0)\Pr(R = 0)$

- R packages developed for data MNAR

  - missingHE
  - miceMANR with paper

Considerations for fairness in missing data imputation

- MI is a powerful tool that can reduce bias and improve efficiency when data is MCAR/MAR
- However, the results from MI is "only as good as" the data that we provide in the imputation model
- For example, if we have extremely few observations from a certain subgroup, then we have less information on this subgroup even if the rate of missingness is the same across all subgroups
- Imbalance of missingness could also affect imputation fairness, e.g., more missingness one group has, the larger the imputation error
- Imputation fairness is associated with missing rate
- Use imputed data for prediction: trade-off between fairness and prediction performance

  - simulation studies have shown that increasing fairness in imputation doesn't lead to more accurate predictions but the opposite results are commonly observed