Multiple Imputation in R

Alexander McLain

Contents

1 Multiple Imputation using `mice`

The `mice` package implements a method to deal with missing data. The package creates multiple imputations (replacement values) for multivariate missing data. The method is based on Fully Conditional Specification, where each incomplete variable is imputed by a separate model.

The Multivariate Imputation by Chained Equations (MICE) algorithm can impute mixes of continuous, binary, unordered categorical and ordered categorical data. In addition, MICE can impute continuous two-level data, and maintain consistency between imputations by means of passive imputation. Many diagnostic plots are implemented to inspect the quality of the imputations.

```
library(mice)
```

Here, we're going to use the `nhanes` dataset in `mice` to demonstrate how imputation works. This dataset has the following variables:

- `age`: Age group (1=20-39, 2=40-59, 3=60+)
- `bmi`: Body mass index (kg/m**2)
- `hyp`: Hypertensive (1=no,2=yes)
- `chl`: Total serum cholesterol (mg/dL)

Let's look at the data and examine the missingness patterns.

```
head(nhanes)
```

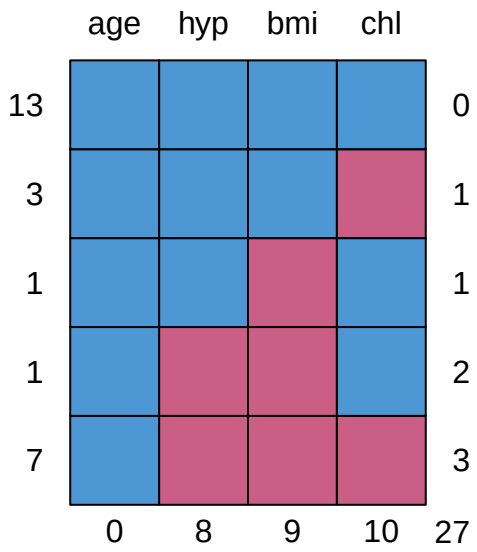
```
##   age  bmi hyp chl
## 1   1   NA  NA  NA
## 2   2 22.7   1 187
## 3   1   NA   1 187
## 4   3   NA  NA  NA
## 5   1 20.4   1 113
## 6   3   NA  NA 184
```

```
summary(nhanes)
```

```
##       age            bmi             hyp             chl
##  Min.   :1.00   Min.   :20.40   Min.   :1.000   Min.   :113.0
##  1st Qu.:1.00   1st Qu.:22.65   1st Qu.:1.000   1st Qu.:185.0
```

```
##  Median :2.00   Median :26.75   Median :1.000   Median :187.0
##  Mean   :1.76   Mean   :26.56   Mean   :1.235   Mean   :191.4
##  3rd Qu.:2.00   3rd Qu.:28.93   3rd Qu.:1.000   3rd Qu.:212.0
##  Max.   :3.00   Max.   :35.30   Max.   :2.000   Max.   :284.0
##                 NA's   :9       NA's   :8       NA's   :10
```

```
md.pattern(nhanes)
```




```
##    age hyp bmi chl
## 13   1   1   1   1  0
## 3    1   1   1   0  1
## 1    1   1   0   1  1
## 1    1   0   0   1  2
## 7    1   0   0   0  3
##      0   8   9  10 27
```

1.1 Basic Imputation

```
imp <- mice(nhanes)
```

```
##
##  iter imp variable
##   1   1  bmi  hyp  chl
##   1   2  bmi  hyp  chl
##   1   3  bmi  hyp  chl
##   1   4  bmi  hyp  chl
##   1   5  bmi  hyp  chl
##   2   1  bmi  hyp  chl
##   2   2  bmi  hyp  chl
##   2   3  bmi  hyp  chl
##   2   4  bmi  hyp  chl
##   2   5  bmi  hyp  chl
##   3   1  bmi  hyp  chl
##   3   2  bmi  hyp  chl
##   3   3  bmi  hyp  chl
##   3   4  bmi  hyp  chl
##   3   5  bmi  hyp  chl
```

```
##   4   1  bmi  hyp  chl
##   4   2  bmi  hyp  chl
##   4   3  bmi  hyp  chl
##   4   4  bmi  hyp  chl
##   4   5  bmi  hyp  chl
##   5   1  bmi  hyp  chl
##   5   2  bmi  hyp  chl
##   5   3  bmi  hyp  chl
##   5   4  bmi  hyp  chl
##   5   5  bmi  hyp  chl
```

```
imp
```

```
## Class: mids
## Number of multiple imputations:  5
## Imputation methods:
##   age   bmi   hyp   chl
##    "" "pmm" "pmm" "pmm"
## PredictorMatrix:
##     age bmi hyp chl
## age   0   1   1   1
## bmi   1   0   1   1
## hyp   1   1   0   1
## chl   1   1   1   0
```

To look at the imputated data we use

```
imp$imp
```

```
## $age
## [1] 1 2 3 4 5
## <0 rows> (or 0-length row.names)
##
## $bmi
##       1    2    3    4    5
## 1  26.3 33.2 29.6 22.5 30.1
## 3  22.0 33.2 27.2 25.5 28.7
## 4  20.4 27.2 20.4 27.4 20.4
## 6  20.4 25.5 21.7 22.7 24.9
## 10 29.6 27.4 27.5 26.3 26.3
## 11 29.6 29.6 30.1 22.0 28.7
## 12 30.1 26.3 22.7 22.0 30.1
## 16 29.6 35.3 29.6 25.5 30.1
## 21 33.2 33.2 30.1 22.5 27.2
##
## $hyp
##    1 2 3 4 5
## 1  1 1 1 1 1
## 4  2 2 1 2 2
## 6  2 1 1 1 1
## 10 2 1 1 2 1
## 11 1 1 1 1 1
## 12 1 1 1 2 2
## 16 1 1 1 1 1
## 21 1 1 1 1 1
##
```

```
## $chl
##      1   2   3   4   5
## 1  187 229 238 118 238
## 4  184 204 218 218 218
## 10 218 187 218 204 187
## 11 187 238 238 238 187
## 12 204 206 229 186 199
## 15 187 229 131 187 131
## 16 187 229 187 238 187
## 20 284 284 184 284 204
## 21 186 131 238 238 131
## 24 218 204 206 218 186
```

If we wanted to imput more data we use the `m=` arguement. Here we'll supress the output.

```
imp <- mice(nhanes, m = 20, print = F)
```

1.2 Setting the seed

The algorithm uses random sampling, and therefore, the results will be (perhaps slightly) different if we repeat the imputations with different seeds. In order to get exactly the same result, use the seed argument

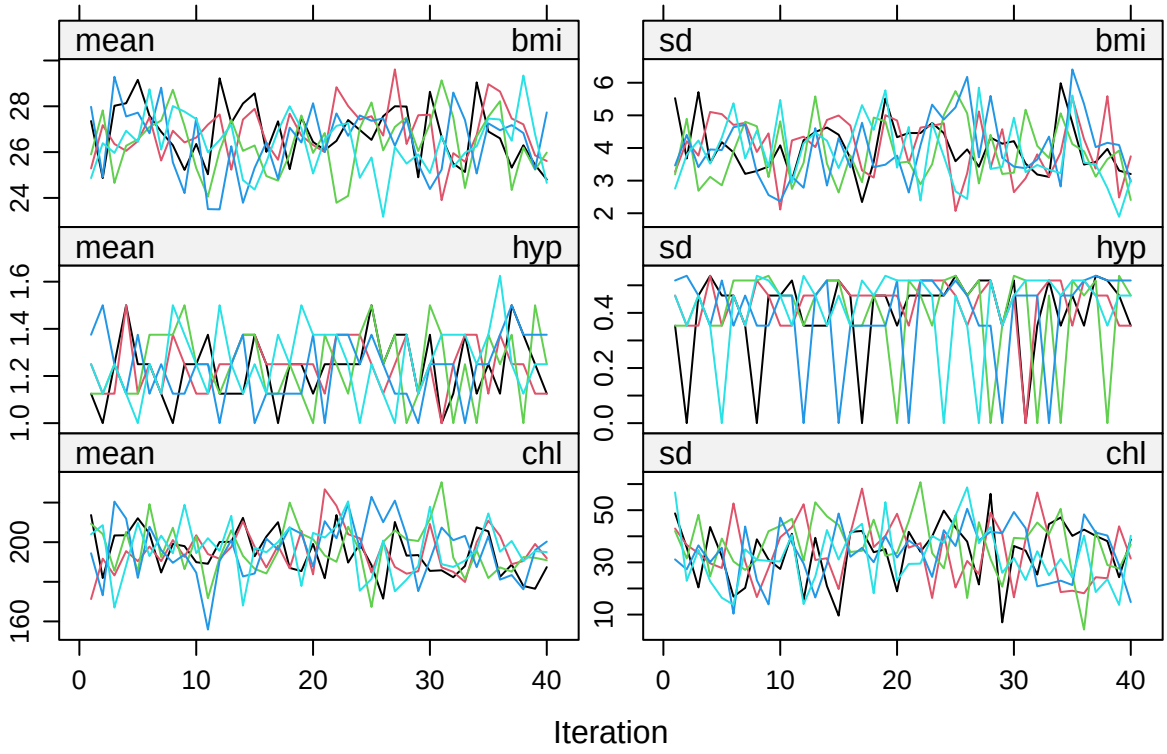
```
imp <- mice(nhanes, seed=123, print=F)
```

where 123 is some arbitrary number that you can choose yourself. Rerunning this command will always yields the same imputed values.

1.3 Running additional iterations

Though using just five iterations (the default) often works well in practice, we need to extend the number of iterations of the mice algorithm to confirm that there is no trend and that the trace lines intermingle well. We can increase the number of iterations to 40 by running 35 additional iterations using the `mice.mids()` function.

```
imp40 <- mice.mids(imp, maxit=35, print=F)
plot(imp40)
```



1.4 Linear regression analysis of the data

Now let's do an analysis of the data.

```
fit <- with(imp, lm(bmi ~ chl))
fit
```

```
## call :
## with.mids(data = imp, expr = lm(bmi ~ chl))
##
## call1 :
## mice(data = nhanes, printFlag = F, seed = 123)
##
## nmis :
## age bmi hyp chl
##   0   9   8  10
##
## analyses :
## [[1]]
##
## Call:
## lm(formula = bmi ~ chl)
##
## Coefficients:
## (Intercept)          chl
##   26.004563     0.007469
##
##
## [[2]]
##
```

```
## Call:
## lm(formula = bmi ~ chl)
##
## Coefficients:
## (Intercept)          chl
##    18.84425      0.04025
##
##
## [[3]]
##
## Call:
## lm(formula = bmi ~ chl)
##
## Coefficients:
## (Intercept)          chl
##    22.37823      0.02214
##
##
## [[4]]
##
## Call:
## lm(formula = bmi ~ chl)
##
## Coefficients:
## (Intercept)          chl
##    18.29446      0.04629
##
##
## [[5]]
##
## Call:
## lm(formula = bmi ~ chl)
##
## Coefficients:
## (Intercept)          chl
##    20.09149      0.03235
```

```
pool.fit <- pool(fit)
summary(pool.fit)
```

```
##          term    estimate std.error statistic       df     p.value
## 1 (Intercept) 21.12259925 5.3445795  3.952154 8.057347 0.004163563
## 2         chl  0.02970031 0.0266314  1.115236 8.447134 0.295469810
```

1.5 Analyzing a created new variables

In all, let's say we want to 20 imputed datasets with 40 iterations while setting the seed. We would run,

```
imp <- mice(nhanes, m=20, maxit = 35, seed=123, print=F)
```

Notice that the model above used a variable that was already in the dataset. How would we create a variable and analyze it? In lm, this is actually simple. We can just create the variable in the formula statement.

For example, let's say we wanted to use $(bmi - age)/age$ as our outcome. Then we could use:

```
fit2 <- with(imp, lm((bmi-age)/age ~ chl + hyp))
pool.fit2 <- pool(fit2)
summary(pool.fit2)
```

```
##          term    estimate  std.error  statistic       df     p.value
## 1 (Intercept) 35.09012461 9.94422633  3.5286933 17.27147 0.002526952
## 2         chl -0.03351422 0.05537092 -0.6052675 13.81487 0.554814532
## 3         hyp -8.51038804 5.21079648 -1.6332221 13.75342 0.125096907
```