How can I perform multiple imputation on longitudinal data using ICE?

Authored by

stats writer

July 1, 2024

Multiple imputation using the Iterative Chained Equations (ICE) method is a statistical technique used to handle missing data in longitudinal studies. This approach involves creating multiple imputed datasets by randomly filling in the missing values based on a set of predictive equations. The process is repeated several times, with each iteration improving the imputed values until a satisfactory level of convergence is achieved. The resulting imputed datasets can then be used for further analysis, taking into account the uncertainty introduced by the missing data. This method allows for a more robust and accurate analysis of longitudinal data by reducing bias and increasing the precision of estimates.

How can I perform multiple imputation on longitudinal data using ICE? | Stata FAQ

How can I perform multiple imputation on longitudinal data using ICE?

Imputing longitudinal or panel data poses special problems. If the data are in long form, each case has multiple rows in the dataset, so this needs to be accounted for in the estimation of any analytic model. At the same time, the information from other time points can be important predictors of missing values, so we want to take advantage of this and incorporate this into our imputation model. The following example shows how to impute longitudinal data, accommodating the structure of this type of data. The example dataset contains data on student's reading and

math scores at
three time points (read and math respectively),
as well as data on the time invariant covariates female, private, and ses. The
data are in long form, so there are 3 rows in the data for each of the 200 students
for whom we have
data. The data also contain an id variable, which allows us to match the cases across the three waves of data collection, and a variable
time which tells us when the data were collected.
There are missing data on three of the four substantive variables. This FAQ page will address the following questions:
(1) How does one create multiple imputed datasets that account for the clustering in the data (multiple observations per student); (2) How does one take advantage of the fact that reading or math scores at the other two time points are likely to be good predictors of any missing values
of the time-varying variables?

First we want to look at our data to confirm that there is missing data, we

can do this using the summarize command (which can be abbreviated to sum). We can also use the user-written command nmissing to look at the amount of missingness per variable within our data. You can download nmissing from within Stata by typing search nmissing (see How can I use the search command to search for programs and get additional help? for more information about using search).

```
use "https://stats.idre.ucla.edu/stat/stata/faq/mi_longi.dta"

sum female ses private read math

Variable | Obs Mean Std. Dev. Min Max
-------------+--------------------------------------------------------
female | 600 .545 .4983864 0 1
ses | 531 2.011299 .7136568 1 3
private | 570 .1526316 .3599479 0 1
read | 538 52.66543 10.29398 26 76
math | 554 52.24676 9.339812 26 75
```

nmissing

ses 69
read 62
math 46
private 30

Stata has a suite of multiple imputation (mi) commands to help users not only impute their data but also explore the patterns of missingness present in the data.

In order to use these commands the dataset in memory must be declared or mi set as "mi" dataset. A dataset that is mi set is given an mi style. This tells Stata how the multiply imputed data is to be stored once the imputation has been completed. For information on these style type help mi styles into the command window. We will use the style mlong. The chosen style can be changed using mi convert.

mi set mlong

You will notice that executing the previous comand will create three new variables to your dataset. These new

variables will be used by Stata to track the imputed datasets and values.

The mi misstable commands helps users tabulate the amount of missing in their variables of interest (summarize) as well as examine patterns of missing(patterns).

mi misstable summarize female private ses read math
Obs<.
+-----------------------------
| | Unique
Variable | Obs=. Obs>. Obs<. | values Min Max
-------------+-----------------------------+-----------------------------
private | 30 570 | 2 0 1
ses | 69 531 | 3 1 3
read | 62 538 | 42 26 76
math | 46 554 | 225 26 75
--
mi misstable patterns female private ses read math
Missing-value patterns
(1 means complete)

| Pattern

Percent | 1 2 3 4

------------+------------

70% | 1 1 1 1

|

9 | 1 1 1 0

8 | 1 1 0 1

6 | 1 0 1 1

4 | 0 1 1 1

2 | 1 1 0 0

<1 | 1 0 0 1

<1 | 0 0 1 1

<1 | 0 1 0 0

<1 | 1 0 1 0

<1 | 0 1 0 1

<1 | 0 1 1 0

<1 | 1 0 0 0

------------+------------

100% |

Variables are (1) private (2) math (3) read (4) ses

Once we are familiar with our data, the first step in the imputation process is to reshape
the data from long to wide. Having the

**data in wide form takes care of both the nesting issue (there is now only one
row of data per student) and allows us to easily use variables from the other
time periods as predictors of missing values, since in wide form, they are just
other variables in the dataset (rather than being part of another row in the dataset).
We do this using the mi reshape command, and then check the output from reshape to
make sure everything went the way it should, and it has. This version of reshape maintains the structure of the multiply imputed dataset as we switch between wide and long. Note that the variable time is dropped, and that there are now three read variables and three math variables after we reshape.**

```
mi reshape wide read math, i(id) j(time)
reshaping m=0 data ...
(note: j = 1 2 3)

Data long -> wide
-----------------------------------------------------------------------------
Number of obs. 600 -> 200
```

Number of variables 7 -> 10
j variable (3 values) time -> (dropped)
xij variables:
read -> read1 read2 read3
math -> math1 math2 math3

After reshaping the data, and checking to make sure that the reshape command worked as we want it to, we can do whatever steps are necessary to impute the missing values. The important point is that since our data are in wide (rather than long) format, the fact that data are longitudinal does not create any additional complications.

After the data is mi set, Stata requires 3 additional commands to complete our analysis. The first is mi register imputed. This command identifies which variables in the imputation model have missing information.

mi register imputed private ses read1 read2 read3 math1 math2 math3

The second command is mi impute chained where the user specifies the imputation model to be used and the number of imputed data sets to be created. Within this command we can specify a particular distribution to impute each variable under. The chosen imputation method is listed with parentheses directly preceding the variable(s) to which this distribution applies. Note that we also use rseed to set the seed for the random number generator, this will enable you to reproduce the results of our imputation.

On the mi impute chained command line we can use the add option to specify the number of imputations to be performed. In this example we chose 10 imputations. Note, the chosen number of 10 imputation is just for illustrative purposes, your data may require more for valid estimation. Variables on the left side of the equal sign have missing information, while the right side is reserved for variables with no missing information and are therefore solely considered "predictors" of missing values.

mi impute chained (logit) private (ologit) ses (regress) read1 read2 read3 math1 math2 math3 = female,

///add(10) rseed (091107)

math1: regress math1 read1 i.private math2 math3 i.ses read3 read2 female
read1: regress read1 math1 i.private math2 math3 i.ses read3 read2 female
private: logit private math1 read1 math2 math3 i.ses read3 read2 female
math2: regress math2 math1 read1 i.private math3 i.ses read3 read2 female
math3: regress math3 math1 read1 i.private math2 i.ses read3 read2 female
ses: ologit ses math1 read1 i.private math2 math3 read3 read2 female
read3: regress read3 math1 read1 i.private math2 math3 i.ses read2 female
read2: regress read2 math1 read1 i.private math2 math3 i.ses read3 female

Performing chained iterations ...

Multivariate imputation Imputations = 10
Chained equations added = 10
Imputed: m=1 through m=10 updated = 0

Initialization: monotone Iterations = 100

burn-in = 10

private: logistic regression

ses: ordered logistic regression

read1: linear regression

read2: linear regression

read3: linear regression

math1: linear regression

math2: linear regression

math3: linear regression

| Observations per m

|----------

Variable | Complete Incomplete Imputed | Total

----------+----------+----------

private | 190 10 10 | 200

ses | 177 23 23 | 200

read1 | 194 6 6 | 200

read2 | 168 32 32 | 200

read3 | 176 24 24 | 200

math1 | 195 5 5 | 200

math2 | 180 20 20 | 200

math3 | 179 21 21 | 200

(complete + incomplete = total; imputed is the minimum across m
of the number of filled-in observations.)

So now we have our multiply imputed data, but they are still in wide format,
and we will probably want them in long form to run the analyses. We can again use mi reshape, to reshape the data back to long

mi reshape long read math, i(id) j(time)

reshaping m=0 data ...
(note: j = 1 2 3)

Data wide -> long

Number of obs. 200 -> 600
Number of variables 10 -> 7
j variable (3 values) -> time
xij variables:
read1 read2 read3 -> read

math1 math2 math3 -> math

---

reshaping m=1 data ...

reshaping m=2 data ...

reshaping m=3 data ...

reshaping m=4 data ...

reshaping m=5 data ...

reshaping m=6 data ...

reshaping m=7 data ...

reshaping m=8 data ...

reshaping m=9 data ...

reshaping m=10 data ...

assembling results ...

After reshaping the data, we will want to explore our

imputations. It is important to make sure that the imputed values make sense, that they are not out of range of the original values, etc. We can start by summarizing our data, we may also want to use by to look at the values generated by each imputation separately. It might also be useful to generate either boxplots or histograms of our variables, so see that the distributions look reasonable after imputation.

sum female private ses read math

Variable | Obs Mean Std. Dev. Min Max

----------+---

female | 2,430 .5296296 .499224 0 1

private | 2,400 .1725 .3778932 0 1

ses | 2,361 1.991529 .7415714 1 3

read | 2,368 52.87335 10.31176 17.72421 81.7084

math | 2,384 52.43168 9.857669 26 75

Once we have carefully checked our data to make sure there were no problems

in the imputation, we can run an analysis on our data. The third step is mi estimate which runs the analytic model of interest within each of the imputed datasets. It

also combines all the estimates (coefficients and standard errors) across all the imputed datasets to privide us with one set of estimates.
Below we have used
the mi estimate prefix with the command xtreg to predict reading test scores
using time, math test scores (math), and gender (female) accounting for the fact that there are multiple observations per student. The command syntax is the same as for xtreg all
that needs to be added is the mi estimate prefix.

mi estimate: xtreg read math time female, i(id)

Multiple-imputation estimates Imputations = 10
Random-effects GLS regression Number of obs = 600

Group variable: id Number of groups = 200
Obs per group:
min = 3
avg = 3.0
max = 3
Average RVI = 0.0900
Largest FMI = 0.1563
DF adjustment: Large sample DF: min = 389.84

avg = 1,816.79

max = 3,299.97

Model F test: Equal FMI F(3, 2669.5) = 49.94

Within VCE type: Conventional Prob > F = 0.0000

---

read | Coef. Std. Err. t P>|t|

---+---

math | .5416812 .0463393 11.69 0.000 .4505749 .6327874

time | .0614452 .3543575 0.17 0.862 -.6333505 .7562409

female | 2.361504 .9042511 2.61 0.009 .5885544 4.134454

_cons | 22.71853 2.663597 8.53 0.000 17.48349 27.95358

---+---

sigma_u | 4.4672309

sigma_e | 6.5185302

rho | .31956744 (fraction of variance due to u_i)

Note: sigma_u and sigma_e are combined in the original metric.

References

Allison, Paul (2001) Missing Data. Sage University Paperback 136. Sage

Publications: Thousand Oaks, CA. pg 73-75.