THE STATA JOURNAL

Multiple imputation of missing values: Update of ice

Patrick Royston
Cancer Group
MRC Clinical Trials Unit
222 Euston Road
London NW1 2DA
UK

1   Introduction

Royston (2004) introduced `mvis`, an implementation for Stata of MICE, a method of multiple multivariate imputation of missing values under missing-at-random (MAR) assumptions. In a second article, Royston (2005) described `ice`, an upgrade incorporating various improvements and changes to the software based on personal experience, discussion with colleagues, and user requests. This article describes an update to `ice`. The changes are less substantial but nevertheless important enough to warrant a brief explanation. The major modification is that the default method of imputing missing values in `ice` is now by sampling from the posterior predictive distribution rather than by predicted mean matching.

The `ice` system comprises five ado-files: `ice`, `micombine`, `mijoin`, `misplit`, and `uvis`. The last three programs have not been changed and are included in the present release for the sake of completeness.

2   Syntax

`ice` *mainvarlist* `using` *filename* $\big[\,$*if*$\,\big]$ $\big[\,$*in*$\,\big]$ $\big[\,$*weight*$\,\big]$ $\big[\,$ `,` <u>bo</u>ot$\big[\,$(*varlist*)$\,\big]$

    `cc(`*ccvarlist*`)` <u>cm</u>d`(`*cmdlist*`)` <u>cyc</u>les`(#)` <u>dryr</u>un `eq(`*eqlist*`)` <u>genm</u>iss`(`*string*`)`

    <u>id</u>`(`*string*`)` `m(#)` <u>mat</u>ch$\big[\,$(*varlist*)$\,\big]$ `on(`*varlist*`)` <u>nocons</u>tant <u>nosh</u>oweq

    <u>pass</u>ive`(`*passivelist*`)` `replace` <u>see</u>d`(#)` <u>sub</u>stitute`(`*sublist*`)` <u>trace</u>`(`*filename*`)` $\big]$

`uvis` *regression_cmd yvar xvarlist* $\big[\,$*if*$\,\big]$ $\big[\,$*in*$\,\big]$ $\big[\,$*weight*$\,\big]$ `,` <u>gen</u>`(`*newvarname*`)`

    $\big[\,$<u>nocons</u>tant <u>boo</u>t <u>mat</u>ch `replace` <u>see</u>d`(#)` $\big]$

where *regression_cmd* may be `logistic`, `logit`, `mlogit`, `ologit`, or `regress`. All weight types supported by *regression_cmd* are allowed.

micombine *regression_cmd* [*yvar*] [*covarlist*] [*if*] [*in*] [*weight*] [, br
    <u>nocons</u>tant <u>det</u>ail <u>ef</u>orm(*string*) genxb(*newvarname*) <u>imp</u>id(*varname*) lrr
    <u>obs</u>id(*varname*) *regression_cmd_options*]

where *regression_cmd* may be clogit, cnreg, glm, logistic, logit, mlogit, nbreg,
    ologit, oprobit, poisson, probit, qreg, regress, rreg, stcox, streg, or xtgee.
    All weight types supported by *regression_cmd* are allowed.

mijoin, clear [m(#) <u>imp</u>id(*varname*)]

misplit, clear [m(#) <u>imp</u>id(*varname*)]

3   Options

Only the changes to options are described.

3.1   Options for ice

draw[(*varlist*)] has been replaced with match[(*varlist*)]. match[(*varlist*)] instructs
    that each member of *varlist* be imputed with the match option of uvis. This option
    provides prediction matching for each member of *varlist*. If (*varlist*) is omitted,
    all relevant variables are imputed with the match option of uvis. The default, if
    match() is not specified, is to draw from the posterior predictive distribution of each
    variable requiring imputation.

trace(*filename*) allows one to monitor the convergence of the MICE algorithm. For each
    original variable with missing values, the mean of the imputed values is stored as a
    variable in *filename*, together with the cycle number at which that mean was calcu-
    lated. The results are stored only for the final imputation. For diagnostic purposes,
    it is sensible to run trace() with m(1) and many cycles, such as cycles(100).
    When the run is complete, it is helpful to load *filename* into memory and plot the
    mean for each imputed variable against the cycle number. If necessary, smoothing
    may be applied to clarify any apparent pattern. Convergence is judged to have oc-
    curred when the pattern of the imputed means is random. The number of cycles
    needed for convergence is usually obvious from the appearance of the plot.

3.2   Options for uvis

draw has been replaced with match. match creates imputations by prediction matching.
    The default is to draw imputations at random from the posterior distribution of
    the missing values of *yvar*, conditional on the observed values and the members of
    xvarlist.

4   What is new?

The principal changes to `ice` are as follows:

1. The default method of imputation involves drawing from the posterior predictive distribution.

2. With prediction matching in `uvis`, imputation is made at random among candidate values of *yvar* if more than one observation satisfies the matching criterion. Previously, it was likely that just one value of *yvar* would be selected in this situation, giving inappropriately restricted imputations.

3. When arranging the system of chained equations that is the heart of the MICE algorithm, variables are imputed in order of increasing missingness. The variable with the least missingness is imputed first, followed by that with the second, lowest amount, and so on. This approach may speed up convergence to the conditional distribution for each variable. Previously the order was arbitrary (it was base on the order of variables in *mainvarlist*).

4. `ice` reports the number of observations containing 0, 1, 2, . . . missing values before proceeding with the imputation. Use of the `dryrun` option also gives this report.

5   Example

I will compare in a simple example with artificial data the use of `match` with drawing from the posterior. The dataset `test2.dta` contains $n = 120$ observations on two variables, `x` and `y`, related by the equation

$$y_i = 6x_i + e_i$$

where the errors e_i are normally distributed with mean 0 and variance 1. `y` and `x` are strongly correlated (Pearson $r = .985$). Forty values each of `x` and `y` are deleted completely at random, leaving a dataset in which 40 pairs of values of `x` and `y` are observed and 80 pairs have a missing value of either `x` or `y`.

Figure 1 shows the relationship between `y` and `x` in one imputation using prediction matching with 1, 2, 3, 5, 7, or 10 cycles of the MICE algorithm.
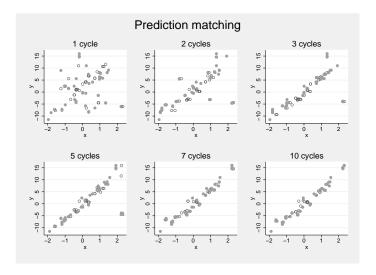
(*Continued on next page*)

Figure 1: Artificial data. Imputation of `x` and `y` using prediction matching after different numbers of cycles of the MICE algorithm. Open circles, `y` missing; filled circles, `x` missing.

The Stata command used was

```
. ice x y using filename1, match(x y) seed(11) trace(filename2) cycles(100) m(1)
```

The open circles show values for which `y` has been imputed, and the filled circles, values for which `x` has been imputed. The 40 pairs in which both `x` and `y` were observed are omitted. The algorithm appears to converge after about 10 cycles. Note the occurrence of "wild" points for small numbers of cycles that are far away from the line $y = 6x$.

Figure 2 shows a trace of the means of `x` and `y` for the first 100 cycles of the MICE algorithm, stored in *filename2*.
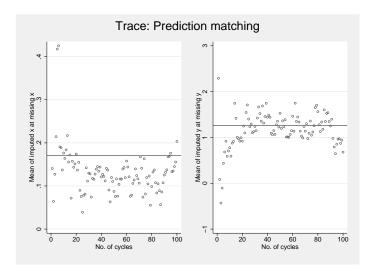
Figure 2: Artificial data. Trace of the first 100 cycles of the MICE algorithm using prediction matching. Horizontal lines, mean of original observations before being set to missing.

The means are initially wild and appear to stabilize after about 20 cycles. The horizontal lines show the mean of the original observations before being set to missing. Most of the imputed means of x are below or substantially below the correct value, suggesting the possibility of bias in the imputation of x in this example. Furthermore, there may be a tendency for the means to form a pattern of oscillation rather than the completely random appearance we would wish for.

Figure 3 repeats figure 1 but using draws from the posterior predictive distributions of x and y instead of prediction matching.
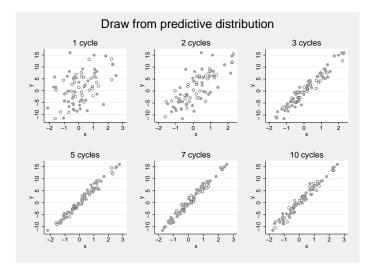
(*Continued on next page*)

Figure 3: Artificial data. Imputation of x and y using draws from the posterior predictive distribution after different numbers of cycles of the MICE algorithm. Open circles, y missing; filled circles, x missing.

The Stata command is the same as before, except that `match(x y)` has been omitted to activate the default drawing algorithm. Two features are apparent. First, the algorithm settles down rapidly and smoothly, with no wild values appearing; the scatter about the line $y = 6x$ is progressively reduced as the number of cycles increases. About five cycles seems enough for convergence. Second, richer sets of imputations are created, since the algorithm is no longer restricted to imputing only observed values of x and y.

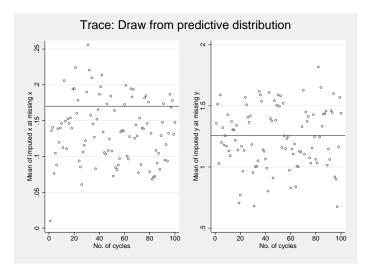Finally, figure 4 repeats figure 2 for the draw method.

Figure 4: Artificial data. Trace of the first 100 cycles of the MICE algorithm using draws from the posterior predictive distributions.

The rapid convergence is clear. The imputed means of `x` now include the correct value. There is no tendency to oscillation.

The example clearly points to the advantages of the draw method when the normality assumptions for continuous variables are fulfilled, as here. The draw method is many times faster than prediction matching.

I now consider in general terms what may be done using data from imputing continuous variables when the normality assumptions fail.

6 Imputing continuous variables

When a continuous variable X has missing values, there are essentially four options for imputing it with `ice`:

1. Assume normality for X and draw from the posterior predictive distribution (the default). Example of `ice` command with `test2.dta`:

   ```
   . ice x y using filename, m(20)
   ```

2. Transform X toward (approximate) normality and draw from the posterior predictive distribution. Retransform back to the original scale. For example,

   ```
   . gen logx=log(x)
   . gen logy=log(y)
   . ice logx logy using filename, m(20)
   . use filename, clear
   ```

```
. replace x=exp(logx)
. replace y=exp(logy)
```

3. Use prediction matching. For example,

```
. ice x y using filename, m(20) match(x y)
```

4. Use ordinal logistic regression (`ologit`). For example,

```
. ice x y using filename, m(20) cmd(ologit)
```

Option 1 is optimal if the normality assumption is (reasonably) appropriate, as in the above example. However, both normality and a continuous distribution for X are assumed. An observed distribution that is heavily grouped or rounded may not give sensible imputed values, since imputations will fall between the observed values. Furthermore, because of the effect of grouping the standard deviation may be incorrect. A possibility is to round the imputed values to resemble the pattern in the observed distribution.

Option 2 should be considered for positively skewed variables; the distribution may often resemble a lognormal. Again, if the original data are grouped, rounding may be considered after transformation back to the original scale. A related possibility is to use the more general Box–Cox transformation to normality (Stata's `boxcox` command).

Option 3 is a reasonable general choice, though concerns exist that prediction matching may give biased imputations, convergence may be slow, and computation may be lengthy (compounded by the need for more MICE cycles).

Option 4 is particularly useful with ordinal variables that either are intrinsically categorical or take a restricted set of values because rounding has been applied. In Stata, the `ologit` command is restricted to response variables with 50 or fewer categories, so variables with more than 50 distinct values will need to be grouped or rounded before imputation is performed.

6.1   Example

As an example of the problems of option 1, imputation with an inappropriate assumption of normality, figure 5 shows the distribution of the variable `x5` (number of positive lymph nodes) in the breast cancer dataset `brcaex.dta` analyzed by Royston (2004).
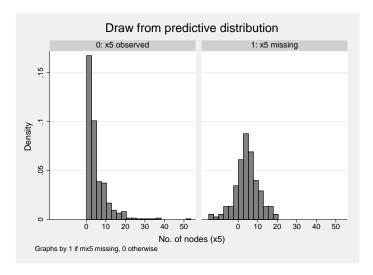
Figure 5: Breast cancer data. Imputation of `x5` by drawing from the posterior predictive distribution, assuming normality. Left panel: distribution of observed `x5`. Right panel: distribution of imputed `x5`.

The distribution of `x5` takes the integers 1, 2, . . . , and is highly positively skewed, with more than 25% of the values being 1. The imputed values are symmetrically distributed about the mean of 5, and many are negative. As an alternative (option 4), figure 6 shows the results of using drawing and `ologit`.
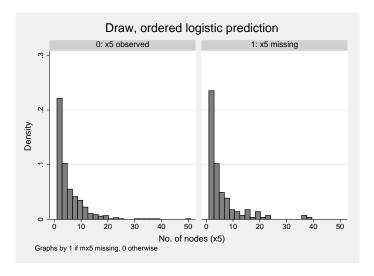

Figure 6: Breast cancer data. Imputation of `x5` by drawing from the posterior predictive distribution, using ordinal logistic regression (`ologit` command). Left panel: distribution of observed `x5`. Right panel: distribution of imputed `x5`.

The distribution of imputed values of `x5` is similar to that of the nonmissing observations—as it should be, given that the missing values were assigned completely at random. The results from prediction matching are much the same as this. The log transformation performs rather less well, although much better than not transforming; the distributional shape does not come out quite right.

7   Conclusion

This paper further develops the MICE software for Stata. It should be seen as work in progress. As experience and knowledge increase, I expect to issue further updates of `ice`.

8   Acknowledgment

I am grateful to Gillian Raab for pointing out issues in the earlier release of `ice` and for providing examples that illustrate some of the problems.

9   References

Royston, P. 2004. Multiple imputation of missing values. *Stata Journal* 4(3): 227–241.

———. 2005. Multiple imputation of missing values: update. *Stata Journal* 5(2): 188–201.

About the Author

Patrick Royston is a medical statistician with 25 years of experience, with a strong interest in biostatistical methodology and in statistical computing and algorithms. He works in clinical trials and related research issues in kidney cancer and other cancers. Currently, he is focusing on problems of model building and validation with survival data, including prognostic factors studies; on complex sample size problems in clinical trials with a survival-time endpoint; on writing a book on multivariable regression modeling; and on new trial designs.