Title

mi replace0 — Replace original data

Syntax

```
mi replace0 using filename, id(varlist)
```

Typical use is

```
. mi extract 0
. (perform data management commands)
. mi replace0 using origfile, id(idvar)
```

Menu

Statistics > Multiple imputation

Description

`mi replace0` replaces $m = 0$ of an `mi` dataset with the non-`mi` data of another and then carries out whatever changes are necessary in $m > 0$ of the former to make the resulting `mi` data consistent. `mi replace0` starts with one of the datasets in memory and the other on disk (it does not matter which is which) and leaves in memory the `mi` data with $m = 0$ replaced.

Option

`id(`*varlist*`)` is required; it specifies the variable or variables to use to match the observations in $m = 0$ of the `mi` data to the observations of the non-`mi` dataset. The ID variables must uniquely identify the observations in each dataset, and equal values across datasets must indicate corresponding observations, but one or both datasets can have observations found (or not found) in the other.

Remarks and examples

It is often easier to perform data management on $m = 0$ and then let `mi replace0` duplicate the results for $m = 1$, $m = 2$, ..., $m = M$ rather than perform the data management on all m's simultaneously. It is easier because $m = 0$ by itself is a non-`mi` dataset, so you can use any of the general Stata commands (that is, non-`mi` commands) with it.

You use `mi extract` to extract $m = 0$; see [MI] **mi extract**. The extracted dataset is just a regular Stata dataset; it is not `mi set`, nor does it have any secret complexities.

You use `mi replace0` to recombine the datasets after you have modified the $m = 0$ data. `mi replace0` can deal with the following changes to $m = 0$:

- changes to the values of existing variables,
- removal of variables,
- addition of new variables,
- dropped observations, and
- added observations.

For instance, you could use `mi extract` and `mi replace0` to do the following:

```
. use my_midata, clear
. mi extract 0
. replace age = 26 if age==6
. replace age = 32 if pid==2088
. merge 1:1 pid using newvars, keep(match) nogen
. by location: egen avgrate = mean(rate)
. drop proxyrate
. mi replace0 using my_midata, id(pid)
```

In the above,

1. we extract $m = 0$;
2. we update existing variable `age` (we fix a typo and the age of `pid` 2088);
3. we merge $m = 0$ with `newvars.dta` to obtain some new variables and, in the process, keep only the observations that were found in both $m = 0$ and `newvars.dta`;
4. we create new variable `avgrate` equal to the mean rate by location; and
5. we drop previously existing variable `proxyrate`.

We then take that result and use it to replace $m = 0$ in our original `mi` dataset. We leave it to `mi replace0` to carry out the changes to $m = 1$, $m = 2$, …, $m = M$ to account for what we did to $m = 0$.

By the way, it turns out that `age` in `my_midata.dta` is registered as imputed. We changed one nonmissing value to another nonmissing value and changed one missing value to a nonmissing value. `mi replace0` will deal with the implications of that. It would even deal with us having changed a nonmissing value to a missing value.

There is no easier way to do data management than by using `mi extract` and `mi replace0`.

Also see

[MI] **intro** — Introduction to mi

[MI] **mi extract** — Extract original or imputed data from mi data