Title

stata.com

mi append — Append mi data

Description

`mi append` is `append` for `mi` data; see [D] **append** for a description of appending datasets.

Menu

Statistics > Multiple imputation

Syntax

`mi append using` *filename* [`,` *options*]

options	Description
`generate(`*newvar*`)`	create *newvar*; 0 = master, 1 = using
`nolabel`	do not copy value labels from using
`nonotes`	do not copy notes from using
`force`	string ↔ numeric not type mismatch error
`noupdate`	see [MI] **noupdate option**

Notes:

1. Jargon:
   master = data in memory
   using = data on disk (*filename*)
2. Master must be `mi set`; using may be `mi set`.
3. `mi append` is logically equivalent to `append`; see [D] **append**. The resulting data have $M = \max(M_{\text{master}}, M_{\text{using}})$, not their sum. See [MI] **mi add** to append imputations holding $m = 0$ constant.
4. `mi append` syntactically differs from `append` in that multiple using files may not be specified and the `keep(`*varlist*`)` option is not allowed.
5. *filename* must be enclosed in double quotes if *filename* contains blanks or other special characters.

Options

`generate(`*newvar*`)` specifies that new variable *newvar* be created containing 0 for observations from the master and 1 for observations from the using.

`nolabel` prevents copying the value-label definitions from the using data to the master. Even if you do not specify this option, label definitions from the using never replace those of the master.

`nonotes` prevents any notes in the using from being incorporated into the master; see [D] **notes**.

`force` allows string variables to be appended to numeric variables and vice versa. The results of such type mismatches are, of course, missing values. Default behavior is to issue an error message rather than append datasets with such violently differing types.

`noupdate` in some cases suppresses the automatic `mi update` this command might perform; see [MI] **noupdate option**.

Remarks and examples

stata.com

Use `mi append` when you would use `append` if the data were not `mi`.

Remarks are presented under the following headings:

Adding new observations

Assume that file `mymi.dta` contains data on three-quarters of the patients in the ICU. The data are `mi set` and $M = 5$. File `remaining.dta` arrives containing the remaining patients. The data are not `mi set`. To combine the datasets, you type

```
. use mymi, clear
. mi append using remaining
```

The original `mi` data had $M = 5$ imputations, and so do the resulting data. If the new data contain no missing values of the imputed variables, you are ready to go. Otherwise, you will need to impute values for the new data.

Adding new observations and imputations

Assume that file `westwing.dta` contains data on patients in the west wing of the ICU. File `eastwing.dta` contains data on patients in the east wing of the ICU. Both datasets are `mi set` with $M = 5$. You originally intended to analyze the datasets separately, but you now wish to combine them. You type

```
. use westwing, clear
. mi append using eastwing
```

The original data had $M = 5$ imputations, and so do the resulting data.

The data for $m = 0$ are the result of running an ordinary `append` on the two $m = 0$ datasets.

The data for $m = 1$ are also the result of running an ordinary `append`, this time on the two $m = 1$ datasets. Thus the result is a combination of observations of `westwing.dta` and `eastwing.dta` in the same way that $m = 0$ is. Imputations for observations that previously existed are obtained from `westwing.dta`, and imputations for the newly appended observations are obtained from `eastwing.dta`.

Adding new observations and imputations, M unequal

Consider the same situation as above, but this time assume $M = 5$ in `westwing.dta` and $M = 4$ in `eastwing.dta`. The combined result will still have $M = 5$. Imputed values in $m = 5$ will be missing for imputed variables from observations in `westwing.dta`.

Treatment of registered variables

It is possible that the two datasets will have variables registered inconsistently.

Variables registered as imputed in either dataset will be registered as imputed in the final result regardless of how they were registered (or unregistered) in the other dataset.

Barring that, variables registered as passive in either dataset will be registered as passive in the final result.

Barring that, variables registered as regular in either dataset will be registered as regular in the final result.

Stored results

`mi append` stores the following in `r()`:

Scalars

`r(N_master)`	number of observations in m=0 in master
`r(N_using)`	number of observations in m=0 in using
`r(M_master)`	number of imputations (M) in master
`r(M_using)`	number of imputations (M) in using

Macros

`r(newvars)`	new variables added

Thus values in the resulting data are

N = # of observations in $m = 0$
= `r(N_master) + r(N_using)`

k = # of variables
= *k_master* + `':word count `r(newvars)''`

M = # of imputations
= `max(r(M_master), r(M_using))`

Also see

[MI] **intro** — Introduction to mi

[D] **append** — Append datasets

[MI] **mi add** — Add imputations from another mi dataset

[MI] **mi merge** — Merge mi data