mi copy — Copy mi flongsep data

Description

`mi copy` *newname* copies flongsep data in memory to *newname* and sets it so that you are working with that copy. *newname* may not be specified with the `.dta` suffix.

In detail, `mi copy` *newname* 1) completes saving the flongsep data to its current name if that is necessary; 2) copies the data to *newname*`.dta`, `_1_`*newname*`.dta`, `_2_`*newname*`.dta`, ..., `_`M`_`*newname*`.dta`; and 3) tells `mi` that you are now working with *newname*`.dta` in memory.

`mi copy` can also be used with wide, mlong, or flong data, although there is no reason you would want to do so. The data are not saved to the original filename as flongsep data would be, but otherwise actions are the same: the data in memory are copied to *newname*`.dta`, and *newname*`.dta` is loaded into memory.

Menu

Statistics > Multiple imputation

Syntax

`mi copy` *newname* [`, replace`]

Option

`replace` specifies that it is okay to overwrite *newname*`.dta`, `_1_`*newname*`.dta`, `_2_`*newname*`.dta`, ..., if they already exist.

Remarks and examples

In Stata, one usually works with a copy of the data in memory. Changes you make to the data are not saved in the underlying disk file until and unless you explicitly save your data. That is not true when working with flongsep data.

Flongsep data are a matched set of datasets, one containing $m = 0$, another containing $m = 1$, and so on. You work with one of them in memory, namely, $m = 0$, but as you work, the other datasets are automatically updated; as you make changes, the datasets on disk change.

Therefore, it is best to work with a copy of your flongsep data and then periodically save the data to the real files, thus mimicking how you work with ordinary Stata datasets. `mi copy` is for just that purpose. After loading your flongsep data, type, for example,

```
. use myflongsep
```

and immediately make a copy,

```
. mi copy newname
```

You are now working with the same data but under a new name. Your original data are safe.

When you reach a point where you would ordinarily save your data, whether under the original name or a different one, type

```
. mi copy original_name_or_different_name, replace
. use newname
```

Later, when you are done with *newname*, you can erase it by typing

```
. mi erase newname
```

Concerning erasure, you will discover that `mi erase` will not let you erase the files when you have one of the files to be erased in memory. Then you will have to type

```
. mi erase newname, clear
```

See [MI] **mi erase** for more information.

For more information on flongsep data, see *Advice for using flongsep* in [MI] **Styles**.

Also see

[MI] **Intro** — Introduction to mi

[MI] **mi erase** — Erase mi datasets

[MI] **Styles** — Dataset styles