

**mi extract** — Extract original or imputed data from mi data

[Description](#)                      [Menu](#)                      [Syntax](#)                      [Options](#)  
[Remarks and examples](#)      [Also see](#)

## Description

`mi extract #` replaces the data in memory with the data for  $m = \#$ . The data are not `mi set`.

## Menu

Statistics > Multiple imputation

## Syntax

```
mi extract # [ , options ]
```

where  $0 \leq \# \leq M$

<i>options</i>	Description
<code>clear</code>	okay to replace unsaved data in memory
<code>esample(...)</code>	rarely specified option
<code>esample(varname)</code>	... syntax when $\# > 0$
<code>esample(varname #<sub>e</sub>)</code>	... syntax when $\# = 0$ ; $1 \leq \#_e \leq M$

## Options

`clear` specifies that it is okay to replace the data in memory even if the current data have not been saved to disk.

`esample(varname [e])` is rarely specified. It is for use after `mi estimate` (see [\[MI\] mi estimate](#)) when the `esample(newvar)` option was specified to store in *newvar* the `e(sample)` for  $m = 1$ ,  $m = 2$ , ...,  $m = M$ . It is now desired to extract the data for one  $m$  and for `e(sample)` set correspondingly.

`mi extract #, esample(varname), # > 0`, is the usual case in this unlikely event. One extracts one of the imputation datasets and redefines `e(sample)` based on the `e(sample)` previously stored for  $m = \#$ .

The odd case is `mi extract 0, esample(varname #e)`, where  $\#_e > 0$ . One extracts the original data but defines `e(sample)` based on the `e(sample)` previously stored for  $m = \#_e$ .

Specifying the `esample()` option changes the sort order of the data.

## Remarks and examples

If you wanted to give up on `mi` and just get your original data back, you could type

```
. mi extract 0
```

You might do this if you wanted to send your original data to a coworker or you wanted to try a different approach to dealing with the missing values in these data. Whatever the reason, the result is that the original data replace the data in memory. The data are not `mi set`. Your original `mi` data remain unchanged.

If you suspected there was something odd about the imputations in  $m = 3$ , you could type

```
. mi extract 3
```

You would then have a dataset in memory that looked just like your original, except the missing values of the imputed and passive variables would be replaced with the imputed and passive values from  $m = 3$ . The data are not `mi set`. Your original data remain unchanged.

## Also see

[MI] [intro](#) — Introduction to `mi`

[MI] [mi replace0](#) — Replace original data