mi import flongsep — Import flongsep-like data into mi

Description

`mi import flongsep` imports flongsep-like data, that is, data in which $m = 0$, $m = 1$, …, $m = M$ are each recorded in separate `.dta` datasets.

`mi import flongsep` converts the data to `mi` flongsep and `mi sets` the data.

Menu

Statistics > Multiple imputation

Syntax

`mi import flongsep` *name*`,` *required_options* [*true_options*]

where *name* is the name of the flongsep data to be created.

required_options	Description
`using(`*filenamelist*`)`	input filenames for $m = 1$, $m = 2$, …
`id(`*varlist*`)`	identifying variable(s)

Note: `use` the input file for $m = 0$ before issuing `mi import flongsep`.

true_options	Description
`imputed(`*varlist*`)`	imputed variables to be registered
`passive(`*varlist*`)`	passive variables to be registered
`clear`	okay to replace unsaved data in memory

Options

`using(`*filenamelist*`)` is required; it specifies the names of the `.dta` datasets containing $m = 1$, $m = 2$, …, $m = M$. The dataset corresponding to $m = 0$ is not specified; it is to be in memory at the time the `mi import flongsep` command is given.

The filenames might be specified as

```
using(ds1 ds2 ds3 ds4 ds5)
```

which states that $m = 1$ is in file `ds1.dta`, $m = 2$ is in file `ds2.dta`, …, and $m = 5$ is in file `ds5.dta`. Also, {#-#} is understood, so the above could just as well be specified as

```
using(ds{1-5})
```

The braced numeric range may appear anywhere in the name, and thus

```
using(ds{1-5}imp)
```

would mean that `ds1imp.dta`, `ds2imp.dta`, …, `ds5imp.dta` contain $m = 1$, $m = 2$, …, $m = 5$.

Alternatively, a comma-separated list can appear inside the braces. Filenames `dsfirstm.dta`, `dssecondm.dta`, …, `dsfifthm.dta` can be specified as

```
using(ds{first,second,third,fourth,fifth}m)
```

Filenames can be specified with or without the `.dta` suffix and may be enclosed in quotes if they contain special characters.

`id(`*varlist*`)` is required; it specifies the variable or variables that uniquely identify the observations in each dataset. The coding must be the same across datasets.

`imputed(`*varlist*`)` and `passive(`*varlist*`)` are truly optional options, although it would be unusual if `imputed()` were not specified.

`imputed(`*varlist*`)` specifies the names of the imputed variables.

`passive(`*varlist*`)` specifies the names of the passive variables.

`clear` specifies that it is okay to replace the data in memory even if they have changed since they were saved to disk.

Remarks and examples

The procedure to convert flongsep-like data to `mi` flongsep is this:

1. `use` the dataset corresponding to $m = 0$.
2. Issue the `mi import flongsep` *name* command, where *name* is the name of the `mi` flongsep data to be created.
3. Perform the checks outlined in *Using mi import nhanes1, ice, flong, and flongsep* of [MI] **mi import**.
4. Use `mi convert` (see [MI] **mi convert**) to convert the data to a more convenient style such as wide, mlong, or flong.

For instance, you have been given the unset datasets `imorig.dta`, `im1.dta`, and `im2.dta`. You are told that these datasets contain the original data and two imputations, that variable `b` is imputed, and that variable `c` is passive and in fact equal to `a` + `b`. Here are the datasets:

```
. use https://www.stata-press.com/data/r19/imorig
. list
```

	subject	a	b	c
1.	101	1	2	3
2.	102	4	.	.

```
. use https://www.stata-press.com/data/r19/im1
```

```
. list
```

	subject	a	b	c
1.	101	1	2	3
2.	102	4	4.5	8.5

```
. save im1
file im1.dta saved
. use https://www.stata-press.com/data/r19/im2
. list
```

	subject	a	b	c
1.	101	1	2	3
2.	102	4	5.5	9.5

```
. save im2
file im2.dta saved
```

These are the same data discussed in [MI] **Styles** but in unset form.

The fact that these datasets are nicely sorted is irrelevant. To import these datasets, you type

```
. use https://www.stata-press.com/data/r19/imorig
. mi import flongsep mymi, using(im1 im2) id(subject) imputed(b) passive(c)
```

We will now perform the checks outlined in *Using mi import nhanes1, ice, flong, and flongsep* of [MI] **mi import**, which are to run `mi describe` and `mi varying` to verify that variables are registered correctly:

```
. mi describe
Style: flongsep mymi
       last mi update 03mar2025 18:20:15, 0 seconds ago
Observations:
   Complete            1
   Incomplete          1  (M = 2 imputations)
   ---------------------
   Total               2
Variables:
   Imputed: 1; b(1)
   Passive: 1; c(1)
   Regular: 0
   System:  2; _mi_id _mi_miss
   (there are 2 unregistered variables; subject a)
. mi varying
            Possible problem   Variable names
-------------------------------------------------------------------------------
         imputed nonvarying:  (none)
         passive nonvarying:  (none)
       unregistered varying:  (none)
 *unregistered super/varying:  (none)
  unregistered super varying:  (none)
-------------------------------------------------------------------------------
 * super/varying means super varying but would be varying if registered as
   imputed; variables vary only where equal to soft missing in m=0.
```

`mi varying` reported no problems. We finally convert to our preferred wide style:

```
. mi convert wide, clear
. list
```

	subject	a	b	c	_mi_miss	_1_b	_1_c	_2_b	_2_c
1.	101	1	2	3	0	2	3	2	3
2.	102	4	.	.	1	4.5	8.5	5.5	9.5

We are done with the converted data in flongsep format, so we will erase the files:

```
. mi erase mymi
(files mymi.dta _1_mymi.dta _2_mymi.dta erased)
```

Also see

[MI] **Intro** — Introduction to mi

[MI] **mi import** — Import data into mi