mi import flong — Import flong-like data into mi

Syntax
mi import flong, required_options [ true_options ]

required_options

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>m(varname)</td>
<td>name of variable containing m</td>
</tr>
<tr>
<td>id(varlist)</td>
<td>identifying variable(s)</td>
</tr>
</tbody>
</table>

true_options

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>imputed(varlist)</td>
<td>imputed variables to be registered</td>
</tr>
<tr>
<td>passive(varlist)</td>
<td>passive variables to be registered</td>
</tr>
<tr>
<td>clear</td>
<td>okay to replace unsaved data in memory</td>
</tr>
</tbody>
</table>

Menu
Statistics > Multiple imputation

Description
mi import flong imports flong-like data, that is, data in which \( m = 0, m = 1, \ldots, m = M \) are all recorded in one .dta dataset.

mi import flong converts the data to mi flong style. The data are mi set.

Options
m(varname) and id(varlist) are required. m(varname) specifies the variable that takes on values 0, 1, \ldots, M, the variable that identifies observations corresponding to \( m = 0, m = 1, \ldots, m = M \). varname = 0 identifies the original data, varname = 1 identifies \( m = 1 \), and so on.

id(varlist) specifies the variable or variables that uniquely identify observations within m().
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, if any.
clear specifies that it is okay to replace the data in memory even if they have changed since they were saved to disk. Remember, mi import flong starts with flong-like data in memory and ends with mi flong data in memory.
Remarks and examples

The procedure to convert flong-like data to mi flong is this:

1. Use the unset data.
2. Issue the `mi import flong` command.
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 or mlong.

For instance, you have the following unset data:

```
use http://www.stata-press.com/data/r13/ourunsetdata
list, separator(2)
m  subject  a  b  c
 1.        0  101  1  2  3
 2.        0  102  4   .   .
 3.        1  101  1  2  3
 4.        1  102  4  4.5  8.5
 5.        2  101  1  2  3
 6.        2  102  4  5.5  9.5
```

You are told that these data contain the original data (m = 0) and two imputations (m = 1 and m = 2), that variable b is imputed, and that variable c is passive and in fact equal to a + b. These are the same data discussed in [MI] *styles* but in unset form.

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

```
.mi import flong, m(m) id(subject) imputed(b) passive(c)
```

These data are short enough that we can list the result:

```
use http://www.stata-press.com/data/r13/ourunsetdata
list, separator(2)
m  subject  a  b  c   _mi_m  _mi_id  _mi_miss
 1.        0  101  1  2  3   0   1   0
 2.        0  102  4   .   .   0   2   1
 3.        1  101  1  2  3   1   1   .
 4.        1  102  4  4.5  8.5  1   2   .
 5.        2  101  1  2  3   2   1   .
 6.        2  102  4  5.5  9.5  2   2   .
```

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: flong
last mi update 07feb2013 12:59:06, 0 seconds ago
Obs.: complete 1
incomplete 1 (M = 2 imputations)
total 2
Vars.: imputed: 1; b(1)
passive: 1; c(1)
regular: 0
system: 3; _mi_m _mi_id _mi_miss
(there are 3 unregistered variables; m subject a)
. mi varying
Possible problem variable names

  imputed nonvarying: (none)
  passive nonvarying: (none)
  unregistered varying: (none)
*unregistered super/varying: (none)
unregistered super varying: m

* super/varying means super varying but would be varying if registered as
  imputed; variables vary only where equal to soft missing in m=0.

We discover that unregistered variable m is super varying (see [MI] Glossary). Here we no longer
need m, so we will drop the variable and rerun mi varying. We will find that there are no remaining
problems, so we will convert our data to our preferred wide style:

. drop m
. 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 convert wide, clear
. list

<table>
<thead>
<tr>
<th>subject</th>
<th>a</th>
<th>b</th>
<th>c</th>
<th>_mi_miss</th>
<th>_1_b</th>
<th>_1_c</th>
<th>_2_b</th>
<th>_2_c</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>101</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>0</td>
<td>2</td>
<td>3</td>
<td>2</td>
</tr>
<tr>
<td>2</td>
<td>102</td>
<td>4</td>
<td>.</td>
<td>.</td>
<td>1</td>
<td>4.5</td>
<td>8.5</td>
<td>5.5</td>
</tr>
</tbody>
</table>

Also see

[MI] intro — Introduction to mi
[MI] mi import — Import data into mi