Title

stata.com

misstable — Tabulate missing values

Syntax

Report counts of missing values

```
misstable summarize [varlist] [if] [in] [, summarize_options]
```

Report pattern of missing values

```
misstable patterns [varlist] [if] [in] [, patterns_options]
```

Present a tree view of the pattern of missing values

```
misstable tree [varlist] [if] [in] [, tree_options]
```

List the nesting rules that describe the missing-value pattern

```
misstable nested [varlist] [if] [in] [, nested_options]
```

summarize_options	Description
`all`	show all variables
`showzeros`	show zeros in table
`generate(`*stub* `[, exok])`	generate missing-value indicators

patterns_options	Description
`asis`	use variables in order given
`frequency`	report frequencies instead of percentages
`exok`	treat `.a`, `.b`, ..., `.z` as nonmissing
`replace`	replace data in memory with dataset of patterns
`clear`	okay to replace even if original unsaved
`bypatterns`	list by patterns rather than by frequency

tree_options	Description
`asis`	use variables in order given
`frequency`	report frequencies instead of percentages
`exok`	treat `.a`, `.b`, ..., `.z` as nonmissing

nested_options	Description
exok	treat .a, .b, ..., .z as nonmissing

In addition, programmer's option nopreserve is allowed with all syntaxes; see [P] **nopreserve option**.

Menu

Statistics > Summaries, tables, and tests > Other tables > Tabulate missing values

Description

misstable makes tables that help you understand the pattern of missing values in your data.

Options

Options are presented under the following headings:

Options for misstable summarize

all specifies that the table should include all the variables specified or all the variables in the dataset. The default is to include only numeric variables that contain missing values.

showzeros specifies that zeros in the table should display as 0 rather than being omitted.

generate(*stub* [, exok]) requests that a missing-value indicator *newvar*, a new binary variable containing 0 for complete observations and 1 for incomplete observations, be generated for every numeric variable in *varlist* containing missing values. If the all option is specified, missing-value indicators are created for all the numeric variables specified or for all the numeric variables in the dataset. If exok is specified within generate(), the extended missing values .a, .b, ..., .z are treated as if they do not designate missing.

For each variable in *varlist*, *newvar* is the corresponding variable name *varname* prefixed with *stub*. If the total length of *stub* and *varname* exceeds 32 characters, *newvar* is abbreviated so that its name does not exceed 32 characters.

Options for misstable patterns

asis, frequency, and exok – see *Common options* below.

replace specifies that the data in memory be replaced with a dataset corresponding to the table just displayed; see *misstable patterns* under *Remarks and examples* below.

clear is for use with replace; it specifies that it is okay to change the data in memory even if they have not been saved to disk.

bypatterns specifies the table be ordered by pattern rather than by frequency. That is, bypatterns specifies that patterns containing one incomplete variable be listed first, followed by those for two incomplete variables, and so on. The default is to list the most frequent pattern first, followed by the next most frequent pattern, etc.

Options for misstable tree

asis, frequency, and exok – see *Common options* below.

Option for misstable nested

exok – see *Common options* below.

Common options

asis specifies that the order of the variables in the table be the same as the order in which they are specified on the misstable command. The default is to order the variables by the number of missing values, and within that, by the amount of overlap of missing values.

frequency specifies that the table should report frequencies instead of percentages.

exok specifies that the extended missing values .a, .b, ..., .z should be treated as if they do not designate missing. Some users use extended missing values to designate values that are missing for a known and valid reason.

nopreserve is a programmer's option allowed with all misstable commands; see [P] **nopreserve option**.

Remarks and examples

stata.com

Remarks are presented under the following headings:

In what follows, we will use data from a 125-observation, fictional, student-satisfaction survey:

```
. use http://www.stata-press.com/data/r13/studentsurvey
(Student Survey)
. summarize
```

Variable	Obs	Mean	Std. Dev.	Min	Max
m1	125	2.456	.8376619	1	4
m2	125	2.472	.8089818	1	4
age	122	18.97541	.8763477	17	21
female	122	.5245902	.5014543	0	1
dept	116	2.491379	1.226488	1	4
offcampus	125	.36	.4819316	0	1
comment	0				

The m1 and m2 variables record the student's satisfaction with teaching and with academics. comment is a string variable recording any comments the student might have had.

misstable summarize

▷ Example 1

`misstable summarize` reports counts of missing values:

```
. misstable summarize
```

				Obs<.		
Variable	Obs=.	Obs>.	Obs<.	Unique values	Min	Max
age	3		122	5	17	21
female	3		122	2	0	1
dept	9		116	4	1	4

Stata provides 27 different missing values, namely, `.`, `.a`, `.b`, ..., `.z`. The first of those, `.`, is often called system missing. The remaining missing values are called extended missings. The nonmissing and missing values are ordered *nonmissing* $<$ `.` $<$ `.a` $<$ `.b` $<\cdots<$ `.z`. Thus reported in the column "Obs=." are counts of system missing values; in the column "Obs>.", extended missing values; and in the column "Obs<.", nonmissing values.

The rightmost portion of the table is included to remind you how the variables are encoded.

Our data contain seven variables and yet `misstable` reported only three of them. The omitted variables contain no missing values or are string variables. Even if we specified the varlist explicitly, those variables would not appear in the table unless we specified the `all` option.

We can also create missing-value indicators for each of the variables above using the `generate()` option:

```
. quietly misstable summarize, generate(miss_)
. describe miss_*
```

variable name	storage type	display format	value label	variable label
miss_age	byte	%8.0g		(age>=.)
miss_female	byte	%8.0g		(female>=.)
miss_dept	byte	%8.0g		(dept>=.)

For each variable containing missing values, the `generate()` option creates a new binary variable containing `0` for complete observations and `1` for incomplete observations. In our example, three new missing-value indicators are generated, one for each of the incomplete variables `age`, `female`, and `dept`. The naming convention of `generate()` is to prefix the corresponding variable names with the specified *stub*, which is `miss_` in this example.

Missing-value indicators are useful, for example, for checking whether data are missing completely at random. They are also often used within the multiple-imputation context to identify the observed and imputed data; see [MI] **intro substantive** for a general introduction to multiple imputation. Within Stata's multiple-imputation commands, an incomplete value is identified by the system missing value, a dot. By default, `misstable summarize, generate()` marks the extended missing values as incomplete values, as well. You can use `exok` within `generate()` to treat extended missing values as complete when creating missing-value identifiers.

◁

misstable patterns

▷ Example 2

misstable patterns reports the pattern of missing values:

```
. misstable patterns
  Missing-value patterns
    (1 means complete)

             |  Pattern
  Percent    |  1  2  3
-------------+---------
      93%    |  1  1  1

       5     |  1  1  0
       2     |  0  0  0
-------------+---------
     100%    |
  Variables are  (1) age  (2) female  (3) dept
```

There are three patterns in these data: (1,1,1), (1,1,0), and (0,0,0). By default, the rows of the table are ordered by frequency. In larger tables that have more patterns, it is sometimes useful to order the rows by pattern. We could have obtained that by typing mi misstable patterns, bypatterns.

In a pattern, 1 indicates that all values of the variable are nonmissing and 0 indicates that all values are missing. Thus pattern (1,1,1) means no missing values, and 93% of our data have that pattern. There are two patterns in which variables are missing, (1,1,0) and (0,0,0). Pattern (1,1,0) means that age is nonmissing, female is nonmissing, and dept is missing. The order of the variables in the patterns appears in the key at the bottom of the table. Five percent of the observations have pattern (1,1,0). The remaining 2% have pattern (0,0,0), meaning that all three variables contain missing.

As with misstable summarize, only numeric variables that contain missing are listed, so had we typed misstable patterns comments age female offcampus dept, we still would have obtained the same table. Variables that are automatically omitted contain no missing values or are string variables.

The variables in the table are ordered from lowest to highest frequency of missing values, although you cannot see that from the information presented in the table. The variables are ordered this way even if you explicitly specify the varlist with a different ordering. Typing misstable patterns dept female age would produce the same table as above. Specify the asis option if you want the variables in the order in which you specify them.

You can obtain a dataset of the patterns by specifying the replace option:

```
. misstable patterns, replace clear
  Missing-value patterns
    (1 means complete)

             |  Pattern
  Percent    |  1  2  3
-------------+---------
      93%    |  1  1  1

       5     |  1  1  0
       2     |  0  0  0
-------------+---------
     100%    |
  Variables are  (1) age  (2) female  (3) dept
(summary data now in memory)
```

```
. list
```

	_freq	age	female	dept
1.	3	0	0	0
2.	6	1	1	0
3.	116	1	1	1

The differences between the dataset and the printed table are that 1) the dataset always records frequency and 2) the rows are reversed.

◁

misstable tree

▷ Example 3

`misstable tree` presents a tree view of the pattern of missing values:

```
. use http://www.stata-press.com/data/r13/studentsurvey, clear
(Student Survey)
. misstable tree, frequency
```

Nested pattern of missing values

dept	age	female
9	3	3
		0
	6	0
		6
116	0	0
		0
	116	0
		116

(number missing listed first)

In this example, we specified the `frequency` option to see the table in frequency rather than percentage terms. In the table, each column sums to the total number of observations in the data, 125. Variables are ordered from those with the most missing values to those with the least. Start with the first column. The `dept` variable is missing in 9 observations and, farther down, the table reports that it is not missing in 116 observations.

Go back to the first row and read across, but only to the second column. The `dept` variable is missing in 9 observations. Within those 9, `age` is missing in 3 of them and is not missing in the remaining 6. Reading down the second column, within the 116 observations that `dept` is not missing, `age` is missing in 0 and not missing in 116.

Reading straight across the first row again, `dept` is missing in 9 observations, and within the 9, `age` is missing in 3, and within the 3, `female` is also missing in 3. Skipping down just a little, within the 6 observations for which `dept` is missing and `age` is not missing, `female` is not missing, too.

◁

misstable nested

▷ Example 4

`misstable nested` lists the nesting rules that describe the missing-value pattern,

```
. misstable nested
     1.  female(3) <-> age(3) -> dept(9)
```

This line says that in observations in which `female` is missing, so is `age` missing, and vice versa, and in observations in which `age` (or `female`) is missing, so is `dept`. The numbers in parentheses are counts of the missing values. The `female` variable happens to be missing in 3 observations, and the same is true for `age`; the `dept` variable is missing in 9 observations. Thus `dept` is missing in the 3 observations for which `age` and `female` are missing, and in 6 more observations, too.

In these data, it turns out that the missing-value pattern can be summarized in one statement. In a larger dataset, you might see something like this:

```
. misstable nested
     1.  female(50) <-> age(50) -> dept(120)
     2.  female(50) -> m1(58)
     3.  offcampus(11)
```

`misstable nested` accounts for every missing value. In the above, in addition to `female <-> age -> dept`, we have that `female -> m1`, and we have `offcampus`, the last all by itself. The last line says that the 11 missing values in `offcampus` are not themselves nested in the missing value of any other variable, nor do they imply the missing values in another variable. In some datasets, all the statements will be of this last form.

In our data, however, we have one statement:

```
. misstable nested
     1.  female(3) <-> age(3) -> dept(9)
```

When the missing-value pattern can be summarized in one `misstable nested` statement, the pattern of missing values in the data is said to be monotone.

◁

Execution time of misstable nested

The execution time of `misstable nested` is affected little by the number of observations but can grow quickly with the number of variables, depending on the fraction of missing values within variable. The execution time of the example above, which has 3 variables containing missing, is instant. In worst-case scenarios, with 500 variables, the time might be 25 seconds; with 1,000 variables, the execution time might be closer to an hour.

In situations where `misstable nested` takes a long time to complete, it will produce thousands of rules that will defy interpretation. A 523-variable dataset we have seen ran in 20 seconds and produced 8,040 rules. Although we spotted a few rules in the output that did not surprise us, such as the year of the date being missing implied that the month and the day were also missing, mostly the output was not helpful.

If you have such a dataset, we recommend you run `misstable` on groups of variables that you have reason to believe the pattern of missing values might be related.

Stored results

`misstable summarize` stores the following values of the last variable summarized in `r()`:

Scalars

`r(N_eq_dot)`	number of observations containing `.`
`r(N_gt_dot)`	number of observations containing `.a`, `.b`, ..., `.z`
`r(N_lt_dot)`	number of observations containing nonmissing
`r(K_uniq)`	number of unique, nonmissing values
`r(min)`	variable's minimum value
`r(max)`	variable's maximum value

Macros

`r(vartype)`	`numeric`, `string`, or `none`

`r(K_uniq)` contains `.` if the number of unique, nonmissing values is greater than 500. `r(vartype)` contains `none` if no variables are summarized, and in that case, the value of the scalars are all set to missing (`.`). Programmers intending to access results after `misstable summarize` should specify the `all` option.

`misstable patterns` stores the following in `r()`:

Scalars

`r(N_complete)`	number of complete observations
`r(N_incomplete)`	number of incomplete observations
`r(K)`	number of patterns

Macros

`r(vars)`	variables used in order presented

`r(N_complete)` and `r(N_incomplete)` are defined with respect to the variables specified if variables were specified and otherwise, defined with respect to all the numeric variables in the dataset. `r(N_complete)` is the number of observations that contain no missing values.

`misstable tree` stores the following in `r()`:

Macros

`r(vars)`	variables used in order presented

`misstable nested` stores the following in `r()`:

Scalars

`r(K)`	number of statements

Macros

`r(stmt1)`	first statement
`r(stmt2)`	second statement
.	.
.	.
`r(stmt‘r(K)’)`	last statement
`r(stmt1wc)`	`r(stmt1)` with missing-value counts
`r(vars)`	variables considered

A statement is encoded "*varname*", "*varname op varname*", or "*varname op varname op varname*", and so on; *op* is either "`->`" or "`<->`".

Also see

[MI] **mi misstable** — Tabulate pattern of missing values

[R] **summarize** — Summary statistics

[R] **tabulate oneway** — One-way table of frequencies

[R] **tabulate twoway** — Two-way table of frequencies