Title

stata.com

version — Version control

Description Syntax Remarks and examples Also see

Description

In syntax 1, Stata's `version` command (see [P] **version**) sets the version before entering Mata. This specifies both the compiler and library versions to be used. Syntax 1 is recommended.

In syntax 2, Mata's `version` command sets the version of the library functions that are to be used. Syntax 2 is rarely used.

Syntax

Syntax 1

```
. version #[.#]

. mata:
: ...
: function name(...)
: {
:         ...
: }
: ...
: end
```

Syntax 2

```
: function name(...)
: {
:         version #[.#]
:         ...
: }
```

Remarks and examples

stata.com

Remarks are presented under the following headings:

Purpose of version control

Mata is under continual development, which means not only that new features are being added but also that old features sometimes change how they work. Old features changing how they work is supposedly an improvement—it generally is—but that also means old programs might stop working or, worse, work differently.

`version` provides the solution.

If you are working interactively, nothing said here matters.

If you use Mata in do-files or ado-files, we recommend that you set `version` before entering Mata.

Recommendations for do-files

The recommendation for do-files that use Mata is the same as for do-files that do not use Mata: specify the version number of the Stata you are using on the top line:

```
begin myfile.do
version 18.0                // (or version 18.5 for StataNow)
...
end myfile.do
```

To determine the number that should appear after `version`, type `about` at the Stata prompt:

```
. about
Stata/SE 18.0
  (output omitted)
```

We are using Stata 18.0. If we were using StataNow, we would instead see `StataNow/SE 18.5`.

Coding `version 18.0` will not benefit us today but, in the future, we will be able to rerun our do-file and obtain the same results.

By the way, a do-file is any file that you intend to execute using Stata's `do` or `run` commands (see [R] **do**), regardless of the file suffix. Many users (us included) save Mata source code in `.mata` files and then type `do` *myfile*`.mata` to compile. `.mata` files are do-files; we include the `version` line:

```
begin myfile.mata
version 18.0                // (or version 18.5 for StataNow)
mata:
...
end
end myfile.mata
```

Recommendations for ado-files

Mata functions may be included in ado-files; see [M-1] **Ado**. In such files, set `version` before entering Mata along with, as usual, setting the version at the top of your program:

begin myfile.ado

```
program myfile
        version 18.0      ← as usual
        ...
end
version 18.0              ← new
mata:
...
end
```

end myfile.ado

In StataNow, we would replace each `version 18.0` statement with `version 18.5`.

Compile-time and run-time versioning

What follows is detail. We recommend always following the recommendations above.

There are actually two version numbers that matter—the version number set at the time of compilation, which affects how the source code is interpreted, and the version of the libraries used to supply subroutines at the time of execution.

The `version` command that we used in the previous sections is in fact Stata's `version` command (see [P] **version**), and it sets both versions:

```
. version 18.0
. mata:
: function example()
: {
:         ...
: }
: end
```

In the above, we compile `example()` by using the version 18.0 syntax of the Mata language, and any functions `example()` calls will be the 18.0 version of those functions. Setting `version 18.0` before entering Mata ensured all of that.

In the following example, we compile using version 18.0 syntax and use version 18.2 functions:

```
. version 18.0
. mata:
: function example()
: {
:         version 18.2
:         ...
: }
: end
```

In the following example, we compile using version 18.2 syntax and use version 18.0 functions:

```
. version 18.2
. mata:
: function example()
: {
:         version 18.0
:         ...
: }
: end
```

It is, however, very rare that you will want to compile and execute at different version levels.

Also see

[M-5] **callersversion( )** — Obtain version number of caller

[M-2] **Intro** — Language definition

Stata, Stata Press, and Mata are registered trademarks of StataCorp LLC. Stata and Stata Press are registered trademarks with the World Intellectual Property Organization of the United Nations. StataNow and NetCourseNow are trademarks of StataCorp LLC. Other brand and product names are registered trademarks or trademarks of their respective companies. Copyright © 1985–2023 StataCorp LLC, College Station, TX, USA. All rights reserved.

For suggested citations, see the FAQ on citing Stata documentation.