Description

In addition to using ado-files and Mata, you can add new commands to Stata by using the C language by following a set of programming conventions and dynamically linking your compiled library into Stata. The `program` command with the `plugin` option finds plugins and loads (dynamically links) them into Stata.

If you are interested in writing plugins for Stata in Java, see [P] **Java plugin**.

Syntax

program *handle*, plugin [using(*filespec*)]

Options

plugin specifies that plugins be found and loaded into Stata.

using(*filespec*) specifies a file, *filespec*, containing the plugin. If you do not specify using(), program assumes that the file is named *handle*.plugin and can be found along the ado-path (see [U] **17.5 Where does Stata look for ado-files?**).

Remarks and examples

Plugins are most useful for methods that require the greatest possible speed and involve heavy looping, recursion, or other computationally demanding approaches. They may also be useful if you have a solution that is already programmed in C.

For complete documentation on plugin programming and loading compiled programs into Stata, see https://www.stata.com/plugins/.

References

Drukker, D. M. 2018a. Programming an estimation command in Stata: Preparing to write a plugin. *The Stata Blog: Not Elsewhere Classified.* https://blog.stata.com/2018/02/15/programming-an-estimation-command-in-stata-preparing-to-write-a-plugin/.

———. 2018b. Programming an estimation command in Stata: Writing a C plugin. *The Stata Blog: Not Elsewhere Classified.* https://blog.stata.com/2018/02/20/programming-an-estimation-command-in-stata-writing-a-c-plugin/.

———. 2018c. Programming an estimation command in Stata: Writing a C++ plugin. *The Stata Blog: Not Elsewhere Classified.* https://blog.stata.com/2018/02/22/programming-an-estimation-command-in-stata-writing-a-c-plugin-2/.

Also see

[P] **Automation** — Automation

[P] **program** — Define and manipulate programs

Mata Reference Manual