Estimation command — How to program an estimation command

Description

Information on programming estimation commands is given in *The Stata Blog*. Below, we reproduce the blog Programming an estimation command in Stata: A map to posted entries (updated 23 February 2018) (Drukker 2016a).

Remarks and examples

I have posted a series of entries about programming an estimation command in Stata. They are best read in order. The comprehensive list below allows you to read them from first to last at your own pace.

1. Programming estimators in Stata: Why you should

   To help you write Stata commands that people want to use, I illustrate how Stata syntax is predictable and give an overview of the estimation–postestimation structure that you will want to emulate in your programs.

2. Programming an estimation command in Stata: Where to store your stuff

   I discuss the difference between scripts and commands, and I introduce some essential programming concepts and constructions that I use to write the scripts and commands.

3. Programming an estimation command in Stata: Global macros versus local macros

   I discuss a pair of examples that illustrate the differences between global macros and local macros.

4. Programming an estimation command in Stata: A first ado-command

   I discuss the code for a simple estimation command to focus on the details of how to implement an estimation command. The command that I discuss estimates the mean by the sample average. I begin by reviewing the formulas and a do-file that implements them. I subsequently introduce ado-file programming and discuss two versions of the command. Along the way, I illustrate some of the postestimation features that work after the command.

5. Programming an estimation command in Stata: Using Stata matrix commands and functions to compute OLS objects

   I present the formulas for computing the ordinary least-squares (OLS) estimator, and I discuss some do-file implementations of them. I discuss the formulas and the computation of independence-based standard errors, robust standard errors, and cluster–robust standard errors. I introduce the Stata matrix commands and matrix functions that I use in ado-commands that I discuss in upcoming posts.

6. Programming an estimation command in Stata: A first command for OLS

   I show how to write a Stata estimation command that implements the OLS estimator by explaining the code.
7. Programming an estimation command in Stata: A better OLS command

I use the `syntax` command to improve the command that implements the OLS estimator that I discussed in Programming an estimation command in Stata: A first command for OLS. I show how to require that all variables be numeric variables and how to make the command accept time-series–operated variables.

8. Programming an estimation command in Stata: Allowing for sample restrictions and factor variables

I modify the OLS command discussed in Programming an estimation command in Stata: A better OLS command to allow for sample restrictions, to handle missing values, to allow for factor variables, and to deal with perfectly collinear variables.

9. Programming an estimation command in Stata: Allowing for options

I make three improvements to the command that implements the OLS estimator that I discussed in Programming an estimation command in Stata: Allowing for sample restrictions and factor variables. First, I allow the user to request a robust estimator of the variance–covariance of the estimator. Second, I allow the user to suppress the constant term. Third, I store the residual degrees of freedom in e(df_r) so that test will use the $t$ or $F$ distribution instead of the normal or chi-squared distribution to compute the $p$-value of Wald tests.

10. Programming an estimation command in Stata: Using a subroutine to parse a complex option

I make two improvements to the command that implements the OLS estimator that I discussed in Programming an estimation command in Stata: Allowing for options. First, I add an option for a cluster–robust estimator of the variance–covariance of the estimator (VCE). Second, I make the command accept the modern syntax for either a robust or a cluster–robust estimator of the VCE. In the process, I use subroutines in my ado-program to facilitate the parsing, and I discuss some advanced parsing tricks.

11. Programming an estimation command in Stata: Mata 101

I introduce Mata, the matrix programming language that is part of Stata.

12. Programming an estimation command in Stata: Mata functions

I show how to write a function in Mata, the matrix programming language that is part of Stata.

13. Programming an estimation command in Stata: A first ado-command using Mata

I discuss a sequence of ado-commands that use Mata to estimate the mean of a variable. The commands illustrate a general structure for Stata–Mata programs.

14. Programming an estimation command in Stata: Computing OLS objects in Mata

I present the formulas for computing the OLS estimator and show how to compute them in Mata. This post is a Mata version of Programming an estimation command in Stata: Using Stata matrix commands and functions to compute OLS objects. I discuss the formulas and the computation of independence-based standard errors, robust standard errors, and cluster–robust standard errors.

15. Programming an estimation command in Stata: An OLS command using Mata

I discuss a command that computes OLS results in Mata, paying special attention to the structure of Stata programs that use Mata work functions.
16. Programming an estimation command in Stata: Adding robust and cluster-robust VCEs to our Mata-based OLS command

I show how to use the undocumented command _vce_parse to parse the options for robust or cluster-robust estimators of the VCE. I then discuss myregress12.ado, which performs its computations in Mata and computes an independent and identically distributed based, a robust, or a cluster-robust estimator of the VCE.

17. Programming an estimation command in Stata: A review of nonlinear optimization using Mata

I review the theory behind nonlinear optimization and get some practice in Mata programming by implementing an optimizer in Mata. This post is designed to help you develop your Mata programming skills and to improve your understanding of how the Mata optimization suites optimize() and moptimize() work.

18. Programming an estimation command in Stata: Using optimize() to estimate Poisson parameters

I show how to use optimize() in Mata to maximize a Poisson log-likelihood function and to obtain estimators of the VCE based on independent and identically dependent observations or on robust methods.

19. Programming an estimation command in Stata: A poisson command using Mata

I discuss mypoisson1, which computes Poisson-regression results in Mata. The code in mypoisson1.ado is remarkably similar to the code in myregress11.ado, which computes OLS results in Mata, as I discussed in Programming an estimation command in Stata: An OLS command using Mata.

20. Programming an estimation command in Stata: Handling factor variables in optimize()

I discuss a method for handling factor variables when performing nonlinear optimization using optimize(). After illustrating the issue caused by factor variables, I present a method and apply it to an example using optimize().

21. Programming an estimation command in Stata: Handling factor variables in a poisson command using Mata

mypoisson2.ado handles factor variables and computes its Poisson-regression results in Mata. I discuss the code for mypoisson2.ado, which I obtained by adding the method for handling factor variables discussed in Programming an estimation command in Stata: Handling factor variables in optimize() to mypoisson1.ado, discussed in Programming an estimation command in Stata: A poisson command using Mata.

22. Programming an estimation command in Stata: Allowing for robust or cluster-robust standard errors in a poisson command using Mata

mypoisson3.ado adds options for a robust or a cluster-robust estimator of the VCE to mypoisson2.ado, which I discussed in Programming an estimation command in Stata: Handling factor variables in a poisson command using Mata. mypoisson3.ado parses the vce() option using the techniques I discussed in Programming an estimation command in Stata: Adding robust and cluster-robust VCEs to our Mata-based OLS command. I show how to use optimize() to compute the robust or cluster-robust VCE.

23. Programming an estimation command in Stata: Adding analytical derivatives to a poisson command using Mata

Using analytically computed derivatives can greatly reduce the time required to solve a nonlinear estimation problem. I show how to use analytically computed derivatives with optimize(), and I discuss mypoisson4.ado, which uses these analytically computed derivatives. Only a few lines of mypoisson4.ado differ from the code for mypoisson3.ado, which I discussed in
Programming an estimation command in Stata: Allowing for robust or cluster–robust standard errors in a poisson command using Mata.

24. Programming an estimation command in Stata: Making predict work

I make predict work after mypoisson5 by writing an ado-command that computes the predictions and by having mypoisson5 store the name of this new ado-command in e(predict).

25. Programming an estimation command in Stata: Certifying your command

Before you use or distribute your estimation command, you should verify that it produces correct results and write a do-file that certifies that it does so. I discuss the processes of verifying and certifying an estimation command, and I present some techniques for writing a do-file that certifies mypoisson5, which I discussed in previous posts.

26. Programming an estimation command in Stata: Nonlinear least-squares estimators

I want to write ado-commands to estimate the parameters of an exponential conditional mean model and probit conditional mean model by nonlinear least squares. Before I can write these commands, I need to show how to trick optimize() into performing the Gauss–Newton algorithm and apply this trick to these two problems.

27. Programming an estimation command in Stata: Consolidating your code

I write ado-commands that estimate the parameters of an exponential conditional mean model and a probit conditional mean model by nonlinear least squares, using the methods that I discussed in Programming an estimation command in Stata: Nonlinear least-squares estimators. These commands will either share lots of code or repeat lots of code, because they are so similar. It is almost always better to share code than to repeat code. Shared code only needs to be changed in one place to add a feature or to fix a problem; repeated code must be changed everywhere. I introduce Mata libraries to share Mata functions across ado-commands, and I introduce wrapper commands to share ado-code.

28. Programming an estimation command in Stata: Writing an estat postestimation command

estat commands display statistics after estimation. Many of these statistics are diagnostics or tests used to evaluate model specification. Some statistics are available after all estimation commands; others are command specific. I illustrate how estat commands work and then show how to write a command-specific estat command for the mypoisson command that I have been developing.

29. Programming an estimation command in Stata: Preparing to write a plugin

Writing a function in another language (like C, C++, or Java) that Stata calls is known as writing a plugin for Stata or as writing a dynamic-link library (DLL) for Stata. I discuss the tradeoffs of writing a plugin or DLL, and I discuss a simple program whose calculations I will replace with plugins in posts 30–32.

30. Programming an estimation command in Stata: Writing a C plugin

I write a plugin in C that implements the calculations performed by mymean_work() in mymean11.ado, discussed in Programming an estimation command in Stata: Preparing to write a plugin. I discuss the plugin code and how to compile it on a Mac, Windows, and Linux.

31. Programming an estimation command in Stata: Writing a C++ plugin

I write a plugin in C++ that implements the calculations performed by mymean_work() in mymean11.ado, discussed in Programming an estimation command in Stata: Preparing to write a plugin. I discuss the plugin code and how to compile it on a Mac, Windows, and Linux.
32. Programming an estimation command in Stata: Writing a Java plugin

I write a plugin in Java that implements the calculations performed by `mymean_work()` in `mymean11.ado`, discussed in Programming an estimation command in Stata: Preparing to write a plugin. I discuss the plugin code and how to compile it using command-line tools that are compatible across platforms.

References


Also see

[U] 18 Programming Stata

[U] 20 Estimation and postestimation commands