Stata Version 11 and "Microeconometrics using Stata"

A. Colin Cameron, and Pravin K. Trivedi

October 22, 2009

Abstract

Microeconometrics using Stata, published in December 2008, was written for Stata version 10.1. Here we consider use of the book using Stata version 11, released in July 2009.

1. Introduction

Microeconometrics using Stata, published in December 2008, was written for Stata version 10.1. Here we consider use of the book using Stata version 11, released in July 2009.

All the book programs will run in Stata 11 as published, because they all include a version control command – `version 10.1`.

If you want to instead run the programs in Stata 11 under version 11, replace the version control command `version 10.1` with `version 11`. Then all the book programs except for two will run without modification. In the next section, we discuss the changes required for these two examples.

In the following section, we discuss how some of the examples can be more simply implemented using commands introduced in Stata 11.

We assume that the user has used the `update` command in Stata, so that the most recent update of Stata version 11 is being used.

2. Running the book programs in Stata 11 with command version 11

It is better to run the programs in Stata 11 under version 11. To do so wherever the command

```
. version 10.1
```

appears, replace it with the command

```
. version 11
```

In that case all the book programs will run in Stata 11 with two exceptions, examples in programs using the `ml` command in chapter 11 (given in file `mus11p1opt.do`). These exceptions arise because the `ml` command is now a front end to the new Mata `moptimize()` optimization function, necessitating minor changes to the `ml` command to be consistent with `moptimize()`.

First, in the example on pages 374-5 that computes robust standard errors, the `ml` command `d1` method should be changed to the new `e1` method. So on page 375 replace

```
. ml model d1 d1poisrob (docvis = private chronic female income), vce(robust)
```

with

```
. ml model e1 d1poisrob (docvis = private chronic female income), vce(robust)
```

The reason for this change is that the `ml` command `d1` method feeds into the new Mata `moptimize()` function `d1` evaluator which computes the gradient. For robust standard error computation we need more, the contribution of each observation to the gradient. The Mata `moptimize()` function `e1` evaluator does this, and so we use corresponding `ml` command `e1` method. Note that no change has been made to program `d1poisrob` (aside from change to `version 11`) and the only change in the `ml model` command is to change `d1` to `e1`. Similarly if method `d2` was used with a robust VCE option we should change the method to `e2`.

Second, in the example on pages 375-6 that uses the `ml` command `d2` method, the option `negh` should be added. So on page 375 replace

```
. ml model d2 d2pois (docvis = private chronic female income)
```

with

```
. ml model d2 d2pois (docvis = private chronic female income), negh
```

The reason for this change is that the `ml` command `d2` method feeds into new Mata `moptimize()` command `d2` evaluator which as default has the user define the Hessian, so in version 11 the `ml` command `d2` method as default has the user define the Hessian. By contrast, previous versions of Stata asked for definition of minus the Hessian. The version 11 option `negh` signals that minus the Hessian has been provided. A similar change needs to be made if the new `ml` command `e2` method was used to enable computation of a robust estimate of the VCE.

3. Improvements to book programs possible using Stata 11

Stata version 11 introduces several new commands and constructs that make it easier to implement some of the methods given in the book.

3.1. Marginal effects

The new Stata `margins` command supplants the Stata `mfx` command and the user-written `margeff` command. The `mfx` and `margeff` comands can still be used, but it is better to use `margins`. The `margins` command is fully supported by Stata and will compute the average marginal effect for many more models than user-written command `margeff`.

For example, consider the chapter 10.6 Poisson regression

```
. poisson docvis private chronic female income, vce(robust)
```

To obtain, respectively, the marginal effect at the mean (MEM), marginal effect at a representative value (MER) and average marginal effect (AME) for all regressors give the commands

```
. margins, dydx(*) atmean                                          //  MEM
. margins, dydx(*) at(private=1 chronic=0 female=1 income=10)  //  MER
. margins, dydx(*)                                                 //  AME
```

The marginal effects computed using the above commands will use calculus methods. For indicator variables it is better to use the finite-difference method (which is the default for `mfx` and `margeff`). To do so using the `margins` command we first identify the variables that are indicator variables using variable name prefix `i.`

```
. poisson docvis i.private i.chronic i.female income, vce(robust)
. margins, dydx(*) atmean                                          //  MEM
. margins, dydx(*) at(private=1 chronic=0 female=1 income=10)  //  MER
. margins, dydx(*)                                                 //  AME
```

These commands yield the same output as given in chapter 10.6 of the book.

3.2. Factor variables

The prefix `i.` used in the preceding example is now part of a broader construct called factor variables. See especially [U] **25 Working with categorical data and factor variables** (pp. 339-357). Factor variables make it much easier to define models based on sets of indicator variables formed from categorical variables, and models with interactions between any combination of sets of indicator variables and continuous variables. This also makes it much easier to perform hypothesis tests of group effects and to compute marginal effects in models with interactions.

For example, consider the page 345 example of computation of the AME when income appears in a cubic polynomial. The commands

```
. quietly poisson docvis private chronic female c.income c.income#c.income ///
>    c.income#c.income#c.income, vce(robust)
. margins, dydx(income)
```

give the same estimate of the AME as on page 345, and additionally give the standard error and associated 95 percent confidence interval for this estimate.

3.3. Generalized method of moments

Version 11 of Stata introduces the `gmm` command to compute generalized method of moment (GMM) estimators. This makes it much easier to code up the nonlinear instrumental variables examples given in the book.

The Mata-code example given on pages 381-3 can now be done using the much simpler Stata code

```
. gmm (docvis - exp({xb:private chronic female income}+{b0})), ///
>  instruments(firmsize chronic female income) onestep
```

An overidentified model is estimated by adding appropriate variables to the instruments list. In that case one can use option `twostep` to obtain more efficient estimates, and then use post-estimation comand `estat overid` to enable a test of overidentifying restrictions.

A similar change can be made for the example on pages 596-7.

References

Cameron, A.C., and P.K. Trivedi (2009), *Microeconometrics using Stata,* College Station (TX), Stata Press.