1 Read this—it will help

Contents

A Complete Stata Documentation Set contains more than 14,000 pages of information in the following manuals:

[GSM]	*Getting Started with Stata for Mac*
[GSU]	*Getting Started with Stata for Unix*
[GSW]	*Getting Started with Stata for Windows*
[U]	*Stata User's Guide*
[R]	*Stata Base Reference Manual*
[BAYES]	*Stata Bayesian Analysis Reference Manual*
[D]	*Stata Data Management Reference Manual*
[ERM]	*Stata Extended Regression Models Reference Manual*
[FMM]	*Stata Finite Mixture Models Reference Manual*
[FN]	*Stata Functions Reference Manual*
[G]	*Stata Graphics Reference Manual*
[IRT]	*Stata Item Response Theory Reference Manual*
[DSGE]	*Stata Linearized Dynamic Stochastic General Equilibrium Reference Manual*
[XT]	*Stata Longitudinal-Data/Panel-Data Reference Manual*
[ME]	*Stata Multilevel Mixed-Effects Reference Manual*
[MI]	*Stata Multiple-Imputation Reference Manual*
[MV]	*Stata Multivariate Statistics Reference Manual*
[PSS]	*Stata Power and Sample-Size Reference Manual*
[P]	*Stata Programming Reference Manual*
[SP]	*Stata Spatial Autoregressive Models Reference Manual*
[SEM]	*Stata Structural Equation Modeling Reference Manual*
[SVY]	*Stata Survey Data Reference Manual*
[ST]	*Stata Survival Analysis Reference Manual*
[TS]	*Stata Time-Series Reference Manual*
[TE]	*Stata Treatment-Effects Reference Manual: Potential Outcomes/Counterfactual Outcomes*
[I]	*Stata Glossary and Index*
[M]	*Mata Reference Manual*

In addition, installation instructions may be found in the *Installation Guide*.

1.1 Getting Started with Stata

There are three *Getting Started* manuals:

[GSM] *Getting Started with Stata for Mac*
[GSU] *Getting Started with Stata for Unix*
[GSW] *Getting Started with Stata for Windows*

1. Learn how to use Stata—read the *Getting Started* (GSM, GSU, or GSW) manual.
2. Now turn to the other manuals; see **[U] 1.2 The User's Guide and the Reference manuals**.

1.2 The User's Guide and the Reference manuals

The *User's Guide* is divided into three sections: *Stata basics*, *Elements of Stata*, and *Advice*. The table of contents lists the chapters within each of these sections. Click on the chapter titles to see the detailed contents of each chapter.

The *Guide* is full of a lot of useful information about Stata; we recommend that you read it. If you only have time, however, to read one or two chapters, then read **[U] 11 Language syntax** and **[U] 12 Data**.

The other manuals are the *Reference* manuals. The Stata *Reference* manuals are each arranged like an encyclopedia—alphabetically. Look at the *Base Reference Manual*. Look under the name of a command. If you do not find the command, look in the subject index in [I] *Stata Glossary and Index*. A few commands are so closely related that they are documented together, such as `ranksum` and `median`, which are both documented in [R] **ranksum**.

Not all the entries in the *Base Reference Manual* are Stata commands; some contain technical information, such as [R] **maximize**, which details Stata's iterative maximization process, or [R] **error messages**, which provides information on error messages and return codes.

Like an encyclopedia, the *Reference* manuals are not designed to be read from cover to cover. When you want to know what a command does, complete with all the details, qualifications, and pitfalls, or when a command produces an unexpected result, read its description. Each entry is written at the level of the command. The descriptions assume that you have little knowledge of Stata's features when they are explaining simple commands, such as those for using and saving data. For more complicated commands, they assume that you have a firm grasp of Stata's other features.

If a Stata command is not in the *Base Reference Manual*, you can find it in one of the other *Reference* manuals. The titles of the manuals indicate the types of commands that they contain. The *Programming Reference Manual*, however, contains commands not only for programming Stata but also for manipulating matrices (not to be confused with the matrix programming language described in the *Mata Reference Manual*).

1.2.1 PDF manuals

Every copy of Stata comes with Stata's complete PDF documentation.

The PDF documentation may be accessed from within Stata by selecting **Help > PDF documentation**. Even more convenient, every help file in Stata links to the equivalent manual entry. If you are reading **help regress**, simply click on [R] **regress** in the `Title` section of the help file to go directly to the [R] **regress** manual entry.

We provide recommended settings for your PDF viewer to optimize it for Stata's documentation at http://www.stata.com/support/faqs/res/documentation.html.

1.2.1.1 Video example

PDF documentation in Stata

1.2.2 Example datasets

Various examples in this manual use what is referred to as the automobile dataset, `auto.dta`. We have created a dataset on the prices, mileages, weights, and other characteristics of 74 automobiles and have saved it in a file called `auto.dta`. (These data originally came from the April 1979 issue of *Consumer Reports* and from the United States Government EPA statistics on fuel consumption; they were compiled and published by Chambers et al. [1983].)

In our examples, you will often see us type

```
. use http://www.stata-press.com/data/r15/auto
```

We include the `auto.dta` file with Stata. If you want to use it from your own computer rather than via the Internet, you can type

```
. sysuse auto
```

See [D] **sysuse**.

You can also access `auto.dta` by selecting **File > Example datasets...**, clicking on *Example datasets installed with Stata*, and clicking on `use` beside the `auto.dta` filename.

There are many other example datasets that ship with Stata or are available over the web. Here is a partial list of the example datasets included with Stata:

`auto.dta`	1978 Automobile Data
`auto2.dta`	1978 Automobile Data
`autornd.dta`	Subset of 1978 Automobile Data
`bplong.dta`	fictional blood pressure data
`bpwide.dta`	fictional blood pressure data
`cancer.dta`	Patient Survival in Drug Trial
`census.dta`	1980 Census data by state
`citytemp.dta`	City Temperature Data
`citytemp4.dta`	City Temperature Data
`educ99gdp.dta`	Education and GDP
`gnp96.dta`	U.S. GNP, 1967–2002
`lifeexp.dta`	Life expectancy, 1998
`network1.dta`	fictional network diagram data
`network1a.dta`	fictional network diagram data
`nlsw88.dta`	U.S. National Longitudinal Study of Young Women (NLSW, 1988 extract)
`nlswide1.dta`	U.S. National Longitudinal Study of Young Women (NLSW, 1988 extract)
`pop2000.dta`	U.S. Census, 2000, extract
`sandstone.dta`	Subsea elevation of Lamont sandstone in an area of Ohio
`sp500.dta`	S&P 500
`surface.dta`	NOAA Sea Surface Temperature
`tsline1.dta`	simulated time-series data
`tsline2.dta`	fictional data on calories consumed
`uslifeexp.dta`	U.S. life expectancy, 1900–1999
`uslifeexp2.dta`	U.S. life expectancy, 1900–1940
`voter.dta`	1992 presidential voter data
`xtline1.dta`	fictional data on calories consumed

All of these datasets may be used or described from the **Example datasets...** menu listing.

Even more example datasets, including most of the datasets used in the reference manuals, are available at the Stata Press website (http://www.stata-press.com/data/). You can download the datasets with your browser, or you can `use` them directly from the Stata command line:

```
. use http://www.stata-press.com/data/r15/nlswork
```

An alternative to the `use` command for these example datasets is `webuse`. For example, typing

```
. webuse nlswork
```

is equivalent to the above `use` command. For more information, see [D] **webuse**.

1.2.2.1 Video example

Example data included with Stata

1.2.3 Cross-referencing

The *Getting Started* manual, the *User's Guide*, and the *Reference* manuals cross-reference each other.

[R] **regress**
[D] **reshape**
[XT] **xtreg**

The first is a reference to the `regress` entry in the *Base Reference Manual*, the second is a reference to the `reshape` entry in the *Data Management Reference Manual*, and the third is a reference to the `xtreg` entry in the *Longitudinal-Data/Panel-Data Reference Manual*.

[GSW] **B Advanced Stata usage**
[GSM] **B Advanced Stata usage**
[GSU] **B Advanced Stata usage**

are instructions to see the appropriate section of the *Getting Started with Stata for Windows*, *Getting Started with Stata for Mac*, or *Getting Started with Stata for Unix* manual.

1.2.4 The index

The *Glossary and Index* contains a combined index for all the manuals.

To find information and commands quickly, you can use Stata's `search` command; see [R] **search**. At the Stata command prompt, type `search geometric mean`. `search` searches Stata's keyword database and the Internet to find more commands and extensions for Stata written by Stata users.

1.2.5 The subject table of contents

A subject table of contents for the *User's Guide* and all the *Reference* manuals except the *Mata Reference Manual* is located in the *Glossary and Index*. This subject table of contents may also be accessed by clicking on **Contents** in the PDF bookmarks.

1.2.6 Typography

We mix the ordinary typeface that you are reading now with a typewriter-style typeface `that looks like this`. When something is printed in the typewriter-style typeface, it means that something is a command or an option—it is something that Stata understands and something that you might actually type into your computer. Differences in typeface are important. If a sentence reads, "You could list the result ...", it is just an English sentence—you *could* list the result, but the sentence provides no clue as to how you might actually do that. On the other hand, if the sentence reads, "You could `list` the result ...", it is telling you much more—you could list the result, and you could do that by using the `list` command.

We will occasionally lapse into periods of inordinate cuteness and write, "We `described` the data and then `listed` the data." You get the idea. `describe` and `list` are Stata commands. We purposely began the previous sentence with a lowercase letter. Because `describe` is a Stata command, it must be typed in lowercase letters. The ordinary rules of capitalization are temporarily suspended in favor of preciseness.

We also mix in words printed in italic type, such as "To perform the rank-sum test, type `ranksum` *varname*`, by(`*groupvar*`)`". Italicized words are not supposed to be typed; instead, you are to substitute another word for them.

We would also like users to note our rule for punctuation of quotes. We follow a rule that is often used in mathematics books and British literature. The punctuation mark at the end of the quote is included in the quote only if it is a part of the quote. For instance, the pleased Stata user said she thought that Stata was a "very powerful program". Another user simply said, "I love Stata."

In this manual, however, there is little dialogue, and we follow this rule to precisely clarify what you are to type, as in, type "`cd c:`". The period is outside the quotation mark because you should not type the period. If we had wanted you to type the period, we would have included two periods at the end of the sentence: one inside the quotation and one outside, as in, type "the orthogonal polynomial operator, `p.`".

We have tried not to violate the other rules of English. If you find such violations, they were unintentional and resulted from our own ignorance or carelessness. We would appreciate hearing about them.

We have heard from Nicholas J. Cox of the Department of Geography at Durham University, UK, and express our appreciation. His efforts have gone far beyond dropping us a note, and there is no way with words that we can fully express our gratitude.

1.2.7 Vignette

If you look, for example, at the entry [R] **brier**, you will see a brief biographical vignette of Glenn Wilson Brier (1913–1998), who did pioneering work on the measures described in that entry. A few such vignettes were added without fanfare in the Stata 8 manuals, just for interest, and many more were added in Stata 9, and even more have been added in each subsequent release. A vignette could often appropriately go in several entries. For example, George E. P. Box deserves to be mentioned in entries other than [TS] **arima**, such as [R] **boxcox**. However, to save space, each vignette is given once only, and an index of all vignettes is given in the *Glossary and Index*.

Most of the vignettes were written by Nicholas J. Cox, Durham University, and were compiled using a wide range of reference books, articles in the literature, Internet sources, and information from individuals. Especially useful were the dictionaries of Upton and Cook (2014) and Everitt and Skrondal (2010) and the compilations of statistical biographies edited by Heyde and Seneta (2001) and Johnson and Kotz (1997). Of these, only the first provides information on people living at the time of publication.

1.3 What's new

This section is intended for users of the previous version of Stata. If you are new to Stata, you may as well skip to [U] **1.3.18 What's more**.

As always, Stata 15 is 100% compatible with the previous releases, but we remind programmers that it is important to put `version 15`, `version 14.1`, or `version 12`, etc., at the top of old do- and ado-files so that they continue to work as you expect. You were supposed to do that when you wrote them, but if you did not, go back and do it now.

We will list all the changes, item by item, but first, here are the highlights.

1.3.1 What's new (highlights)

The highlights of the release are the following:

1. Latent class analysis (LCA)
2. A `bayes` prefix command that can be used in front of many maximum likelihood estimation command
3. Linearized dynamic stochastic general equilibrium (DSGE) models
4. Extended regression models (ERMs) that fit continuous, binary, ordered responses with 1) endogeneity, 2) Heckman-style selection, and 3) treatment effects
5. Dynamic documents combining Markdown with Stata code to produce HTML files
6. Nonlinear mixed-effects models
7. Spatial autoregressive (SAR) models
8. Interval-censored parametric survival-time models
9. Finite mixture models (FMMs)
10. Mixed logit models
11. Nonparametric regression using kernel methods
12. Power analysis for cluster randomized trials and regression models
13. Produce PDF and Word documents
14. Graph color transparency or opacity
15. ICD-10-CM and ICD-10-PCS support
16. Federal Reserve Economic Data support

And that is not all. The following could have been highlights, too.

- multilevel tobit and interval regression
- heteroskedastic regression
- panel data cointegration tests
- threshold regression
- zero-inflated ordered probit
- Poisson with Heckman-style sample selection
- tests for multiple breaks in time series
- stream random numbers

There is more to boot. The above and other changes are covered here. Detailed sections follow the highlights.

Highlight 1. Latent class analysis (LCA)

Stata's `gsem` command now supports LCA, which, depending on the jargon you use, includes latent profile analysis (LPA) and finite mixture models (FMMs).

All of these models use categorical latent variables. Categorical means group. Latent means unobserved. Categorical latent variables can be used to represent consumers with different buying preferences, patients in different risk groups, or schools serving students with different interests. Unobserved are the buying preferences, risk groups, and interests. These unobserved categories are the latent classes, and LCA is used to identify them while accounting for the uncertainty of the recovered groups and used to account for their effects.

LPA is a variation on LCA and is used when the outcome variables are continuous.

FMM is a synonym for LCA to some people, a subset to others, and a superset to even others. In any case, `gsem` now has the features.

We consider FMM to be a subset of LCA. If you simply want to fit finite mixtures of Poisson or linear regression models and the like, you can use our new `gsem` features, but we have another new feature for you: the `fmm:` prefix command, which is *Highlight 9* below.

LCA, LPA, and FMM are now part of Stata's `gsem` command, and that means you can fit regression models and multioutcome path models that allow parameters to vary across latent classes.

For instance, you might have four binary variables that are indicators of latent groups of consumers. If you believed that there are three such groups, you could type

```
. gsem (y1 y2 y3 y4 <- _cons), lclass(Consum 3) logit
```

`y1`, `y2`, `y3`, and `y4` are observed outcome variables. `Consum` is the latent categorical variable that we specified as taking on three values. The result is to fit a model in which `y1`, `y2`, `y3`, and `y4` are determined by unobserved class.

The command fits four logistic regressions, one for each of the `y` variables. That would fit four intercepts. Because of the new `lclass(Consum 3)` option, however, each of the 4 models would be fit with distinct intercepts for each value of `Consum`, meaning 12 intercepts would be fit for the logistic regressions, and that is not all. A multinomial logistic regression would be used to predict `Consum`.

After fitting the model, you can

- use the new `estat lcprob` command to estimate the proportion of consumers belonging to each class;
- use the new `estat lcmean` command to estimate the marginal means of `y1`, `y2`, `y3`, and `y4` in each class (the means are probabilities in this case);
- use the new `estat lcgof` command to evaluate the goodness of fit;
- use the existing `predict` command to obtain predicted probabilities of class membership and predicted values of observed outcome variables.

See [SEM] **intro 2**, [SEM] **gsem lclass options**, [SEM] **estat lcprob**, [SEM] **estat lcmean**, [SEM] **estat lcgof**, and [SEM] **predict after gsem**.

Highlight 2. bayes prefix

The new `bayes:` prefix command lets you fit Bayesian regression models more easily and fit more models. You always could fit a Bayesian linear regression. Now you can fit it by typing

```
. bayes: regress y x1 x2
```

That is convenient. What you could not previously do was fit a Bayesian survival model. Now you can.

```
. bayes: streg x1 x2, distribution(weibull)
```

You can even fit Bayesian multilevel survival models.

```
. bayes: mestreg x1 x2 || id:, distribution(weibull)
```

In this model, random intercepts were added for each value of variable `id`.

You can use the `bayes:` prefix with the following estimation commands:

Command	Purpose
`bayes: betareg`	Beta regression
`bayes: binreg`	Binomial regression
`bayes: biprobit`	Bivariate probit regression
`bayes: clogit`	Conditional logistic regression
`bayes: cloglog`	Conditional log-log regression
`bayes: fracreg`	Fractional response regression
`bayes: glm`	Generalized linear model
`bayes: gnbreg`	Negative binomial regression
`bayes: heckman`	Heckman selection model
`bayes: heckoprobit`	Ordered probit with sample selection
`bayes: heckprobit`	Probit with sample selection
`bayes: hetprobit`	Heteroskedastic probit
`bayes: hetregress`	Heteroskedastic linear regression
`bayes: intreg`	Interval regression
`bayes: logistic`	Logistic regression (odds ratios)
`bayes: logit`	Logistic regression (coefficients)
	Multilevel mixed-effects ...
`bayes: mecloglog`	complementary log-log regression
`bayes: meglm`	generalized linear model
`bayes: meintreg`	interval regression
`bayes: melogit`	logistic regression
`bayes: menbreg`	negative binomial regression
`bayes: meologit`	ordered logistic regression
`bayes: meoprobit`	ordered probit regression
`bayes: mepoisson`	Poisson regression
`bayes: meprobit`	probit regression
`bayes: mestreg`	parametric survival regression
`bayes: metobit`	tobit regression
`bayes: mixed`	linear regression

bayes: mlogit	Multinomial (polytomous) logistic regression
bayes: mprobit	Multinomial probit regression
bayes: mvreg	Multivariate linear regression
bayes: nbreg	Negative binomial regression
bayes: ologit	Ordered logistic regression
bayes: oprobit	Ordered probit regression
bayes: poisson	Poisson regression
bayes: probit	Probit regression
bayes: regress	Linear regression
bayes: streg	Parametric survival regression
bayes: tnbreg	Truncated negative binomial regression
bayes: tobit	Tobit regression
bayes: tpoisson	Truncated Poisson regression
bayes: truncreg	Truncated linear regression
bayes: zinb	Zero-inflated negative binomial regression
bayes: zioprobit	Zero-inflated ordered probit regression
bayes: zip	Zero-inflated Poisson regression

All of Stata's Bayesian features are supported by the new `bayes:` prefix command. You can select from many prior distributions for model parameters or use default priors. You can use the default adaptive Metropolis–Hastings sampling, or Gibbs sampling, or a combination of the two sampling methods, when available. And you can use any other feature included in `bayesmh`. For example, you can change the default prior distributions for the regression coefficients:

```
. bayes, prior({y: x1 x2}, normal(0,4)): regress y x1 x2
```

After estimation, you can use Stata's standard Bayesian postestimation tools such as `bayesgraph` to check convergence, `bayesstats summary` to estimate functions of model parameters, `bayesstats ic` and `bayestest model` to compute Bayes factors and compare Bayesian models, and `bayestest interval` to perform interval hypotheses testing.

See [BAYES] **bayes** and [BAYES] **bayesian estimation**.

Highlight 3. Linearized dynamic stochastic general equilibrium (DSGE) models

Stata now fits linearized DSGE models, which are time-series models used in economics. These models are an alternative to traditional forecasting models. Both attempt to explain aggregate economic phenomena, but DSGE models do this on the basis of models derived from microeconomic theory.

Being based on microeconomic theory means lots of equations. The key feature of these equations is that expectations of future variables affect variables today. This is one feature that distinguishes DSGEs from a vector autoregression or a state-space model. The other feature is that, being derived from theory, the parameters can usually be interpreted in terms of that theory.

You specify the equations with the `dsge` command. Here is a two-equation model:

```
. dsge (p = {beta}*E(f.p) + {kappa}*y) (f.y = {rho}*y, state)
```

`p` is a control variable, and `y` is a state variable in state-space jargon. `f.` is the forward operator. These equation say the following:

1. The control variable `p` depends on `p` in the future plus κ times `y` today.

2. The expected future value of y is ρ times y today. The state option specifies that y is a state variable.

There are three kinds of variables in DSGE models. Control variables and equations such as p have no shocks and are determined by the system of equations. State variables such as y have implied shocks and are predetermined at the beginning of the time period. Shocks are the stochastic errors that drive the system.

In any case, the above dsge command would define a model and fit it.

If we have a theory about the relationship between β and κ, such as they are equal, we could now test it using test.

Postestimation commands estat policy and estat transition report the policy and transition matrices. If you type

```
. estat policy
```

the control variables as a linear function of the state variables will be displayed. If you had five control variables and three state variables, each of the controls would be reported as a linear function of the three states. In the simple example above, the linear function predicting p will be shown as a function of y today.

```
. estat transition
```

reports the transition matrix. Whereas the policy matrix reports p as a function of y, the transition matrix reports how y evolves through time exclusive of p.

You can produce forecasts using Stata's existing forecast command. You can graph impulse–response functions using Stata's existing irf command.

See [DSGE] **intro**.

Highlight 4. Extended regression models (ERMs)

ERMs is our name for regression models that can account for the following:

1. Endogenous covariates
2. Nonrandom treatment assignment
3. Heckman-style endogenous sample selection

The features may be used in any combination. And it has yet another feature:

4. Forbidden regressions

You can fit models with interactions of endogenous covariates with other covariates, exogenous or endogenous, continuous or dummy, and this includes models containing interactions of an endogenous variable with itself—or said another way, with polynomials of endogenous variables!

In the past, you might have used heckman to fit a linear model with endogenous sample selection or ivregress to fit a linear model with an endogenous covariate, and if you had both problems in one dataset, you were out of luck. You can now use the new eregress command to fit a model to account for both:

```
. eregress y x, select(selvar = x z1 y2) endogenous(y2 = x z2)
```

If you instead have endogenous treatment assignment and an endogenous covariate, type

```
. eregress y x, entreat(trtvar = x z1 y2) endogenous(y2 = x z2)
```

There are four ERM commands.

New command	Fits
`eregress`	linear regression
`eintreg`	interval regression, including tobit
`eprobit`	probit binary outcome
`eoprobit`	ordered probit ordered categorical outcome

Notes:

1. If you use the treatment-effect features, use `estat teffects` after model fitting to obtain treatment effects and potential-outcome means.
2. All the standard postestimation commands are available. `predict` provides predicted values. `margins` computes marginal effects and marginal and conditional mean.

Regressors can be exogenous or endogenous.

Endogenous regressors can be continuous, binary, or ordinal.

Treatment can be endogenous or exogenous. The treatment variable can be binary or ordinal, which is to say, treatment can be multivalued.

Endogenous selection can be modeled using probit or tobit.

You can now fit models that were previously unavailable, even if you need only one of the new features, such as

- interval regression with endogenous covariates
- probit regression with a binary endogenous covariate
- probit regression with endogenous ordinal treatment
- ordered probit regression with endogenous treatment
- linear regression with tobit endogenous sample selection

See [ERM] **intro 8** for an overview and see [ERM] **eregress**, [ERM] **eprobit**, [ERM] **eoprobit**, and [ERM] **eintreg**.

Highlight 5. Dynamic documents using Markdown

Markdown is a standard markup language that provides text formatting from plain text input. It was designed to be easily converted into HTML, the language of the web. Stata now supports it.

You can create HTML files from your Stata output, including graphs. You will start with a plain text file containing Markdown-formatted text and dynamic tags specifying instruction to Stata, such as run this regression or produce that graph. You then use the new `dyndoc` command to convert the file to HTML,

Want to produce TEX documents? With the new `dyntext` command, you can produce any text-based document!

See [P] **dyndoc**, [P] **dyntext**, [P] **markdown**, and [P] **dynamic tags**.

Highlight 6. Nonlinear mixed-effects models

Stata now fits nonlinear mixed-effects models, also known as nonlinear multilevel models and nonlinear hierarchical models. These models can be thought of two ways. You can think of them as nonlinear models containing random effects. Or you can think of them as linear mixed-effects models in which some or all fixed and random effects enter nonlinearly. However you think of them, the overall error distribution is assumed to be Gaussian.

These models are popular because some problems are not, says their science, linear in the parameters. These models are popular in population pharmacokinetics, bioassays, and studies of biological and agricultural growth processes. For example, nonlinear mixed-effects models have been used to model drug absorption in the body, intensity of earthquakes, and growth of plants.

The new estimation command is `menl`. It implements the popular-in-practice Lindstrom–Bates algorithm, which is based on the linearization of the nonlinear mean function with respect to fixed and random effects. Both maximum likelihood and restricted maximum-likelihood estimation methods are supported.

`menl` is easy to use. Single equations can be entered directly, such as

```
. menl weight = ({b1}+{U[plant]})/(1+exp(-(age-{b2})/{b3}))
```

which would fit

$$\text{weight} = \frac{b_1 + U_{\text{plant}}}{1 + \exp\{-(\text{age} - b_2)/b_3\}} + \epsilon$$

To be estimated are b_1, b_2, and b_3. U_{plant} is a random intercept for each plant.

`menl` also allows multistage or hierarchical specifications in which parameters of interest can be defined at each level of hierarchy as functions of other model parameters and random effects, such as

```
. menl weight = {phi1:}/(1+exp(-(age-{phi2:})/{phi3:})),
> define(phi1:{b1}+{U1[plant]}) define(phi2:{b2}+{U2[plant]})
> define(phi3:{b3}+{U3[plant]})
```

This is the same model except that b_2 and b_3 are allowed to vary across plants.

Several variance–covariance structures are available to model the dependence of random effects at the same level of hierarchy. If we wanted, we could have put dependence between `U1`, `U2`, and `U3` in the above example.

There is a within-group error in the model, ϵ. Flexible variance–covariance structures are available to model its heteroskedasticity and its within-group dependence. For example, heteroskedasticity can be modeled as a power function of a covariate or even of predicted mean values, and dependence can be modeled using an autoregressive model of any order.

In addition to standard features, postestimation features also include prediction of random effects and their standard errors, prediction of parameters of interest defined in the model as functions of other model parameters and random effects, estimation of the overall within-cluster correlation matrix, and more.

See [ME] **menl** and [ME] **menl postestimation**.

Highlight 7. Spatial autoregressive (SAR) models

Stata now fits SAR models, also known as simultaneous autoregressive models. The new `spregress`, `spivregress`, and `spxtregress` commands allow spatial lags of the dependent variable, spatial lags of the independent variables, and spatial autoregressive errors. Spatial lags are the spatial

analog of time-series lags. Time-series lags are values of variables from recent times. Spatial lags are values from nearby areas.

The models are appropriate for area (also known as areal) data. Observations are called spatial units and might be countries, states, districts, counties, cities, postal codes, or city blocks. Or they might not be geographically based at all. They could be nodes of a social network. Spatial models estimate direct effects—the effects of areas on themselves—and estimate indirect or spillover effects—effects from nearby areas.

Stata provides a suite of commands for working with spatial data and a new [SP] manual to accompany them. When spatial units are geographically based, you can download standard-format shapefiles from the web that defines the map. With a single command, you can make spillover effects proportional to the inverse distance between areas or restrict them to be just from neighboring areas. And you can create your own custom definitions of proximity.

Provided for fitting models are the following:

Command	Description	Equivalent to
`spregress, gs2sls`	GS2SLS	`regress`
`spregress, ml`	maximum likelihood	`regress`
`spivregress`	endogenous regressors	`ivregress`
`spxtregress, fe`	panel-data fixed effects	`xtreg, fe`
`spxtregress, re`	panel-data random effects	`xtreg, re`

See [SP] **intro**.

Highlight 8. Interval-censored parametric survival-time models

Stata's new `stintreg` command joins `streg` for fitting parametric survival models. `stintreg` fits models to interval-censored data. In interval-censored data, the time of failure is not exactly known. What is known, subject by subject, is a time when the subject had not yet failed and a later time when the subject already had failed.

`stintreg` can fit exponential, Weibull, Gompertz, lognormal, loglogistic, and generalized gamma survival-time models. Both proportional-hazards and accelerated failure-time metrics are supported. Features include

- stratified estimation
- flexible modeling of ancillary parameters
- robust, cluster–robust, bootstrap, and jackknife standard errors

Survey-data estimation is supported via the `svy` prefix.

In addition to the usual features, postestimation features also include plots of survivor, hazard, and cumulative hazard functions, prediction of mean and median times, Cox–Snell and martingale-like residuals, and more.

See [ST] **stintreg** for details.

Highlight 9. Finite mixture models (FMMs)

The new `fmm:` prefix command can be used with 17 Stata estimation commands to FMMs. The commands are the following:

Command	Fits
`fmm: betareg`	Beta regression
`fmm: cloglog`	Complementary log-log regression
`fmm: glm`	Generalized linear models
`fmm: intreg`	Interval-censored regression
`fmm: ivregress`	Instrumental-variable regression
`fmm: logit`	Logistic regression
`fmm: mlogit`	Multinomial logistic regression
`fmm: nbreg`	Negative binomial regression
`fmm: ologit`	Ordered logistic regression
`fmm: oprobit`	Ordered probit regression
`fmm: poisson`	Poisson regression
`fmm: probit`	Probit regression
`fmm: regress`	Linear regression
`fmm: streg`	Parametric survival-time regression
`fmm: tobit`	Tobit regression
`fmm: tpoisson`	Truncated Poisson regression
`fmm: truncreg`	Truncated linear regression

`fmm` fits models when the data come from unobserved subpopulations. That is a broad statement and `fmm:` can support it.

The most typical use of `fmm:` is to fit one model and allow the parameters (coefficients, location, variance, scale, etc.) to vary across subpopulations. We will call these unobserved subpopulations classes. Say we are interested in

```
. regress y x1 x2
```

but we believe there are three classes across which the parameters of the model might vary. Even though we have no variable recording the class membership, we can fit

```
. fmm 3: regress y x1 x2
```

Reported will be separate models for each class and a model for predicting membership in them.

`fmm:` can be used with multiple estimation commands simultaneously when the classes might follow different models, such as

```
. fmm: (regress y x1 x2) (poisson y x1 x2 x3)
```

In this two-class example, reported will be a linear regression model for the first class, a Poisson regression for the second, and a model that predicts class membership.

Postestimation commands are available to 1) estimate each class's proportion in the overall population ([FMM] **estat lcprob**); 2) report marginal means of the outcome variables within class ([FMM] **estat lcmean**); and 3) predict probabilities of class membership and predicted outcomes ([FMM] **fmm postestimation**).

See [FMM] **fmm intro**.

Highlight 10. Mixed logit models

Stata fits discrete choice models. Stata 15 will fit them with random coefficients. Discrete choice is another way of saying multinomial or conditional logistic regression. The word "mixed" is used by statisticians whenever some coefficients are random and others are fixed. Ergo, Stata 15 fits mixed logit models.

Random coefficients arise for many reasons, but there is a special reason researchers analyzing discrete choices might be interested in them. Random coefficients are a way around the IIA assumption. If you have a choice among walking, public transportation, or a car and you choose walking, the other two alternatives are irrelevant. Take one of them away, and you would still choose walking. Human beings sometimes violate this assumption, at least judged by their behavior.

Mathematically speaking, IIA makes alternatives independent after conditioning on covariates. If IIA is violated, then the alternatives would be correlated. Random coefficients allow that.

A requirement for fitting random coefficients is that the variable varies across the alternatives. Thus the mixed logit model is often said to incorporate alternative-specific variables.

The new Stata 15 command that fits this is named `asmixlogit`.

The new command also allows the random coefficients to be drawn from different distributions. One might be normal and another log normal. Also supported are multivariate normal, truncated normal, uniform, and triangular distributions.

See [R] **asmixlogit**.

Highlight 11. Nonparametric regression, kernel methods

Stata now fits nonparametric regressions. In these models, you do not specify a functional form. You specify

$$y = g(x_1, x_2, \ldots, x_k) + \epsilon$$

and $g(\cdot)$ is fit. The method does not assume that $g(\cdot)$ is linear; it could just as well be

$$y = \beta_1 x_1 + \beta_2 x_2^2 + \beta_3 x_1 x_2 + \cdots$$

and it does not even assume it is linear in the parameters. It could just as well be

$$y = \beta_1 x_1^{\beta_2} + \beta_3 \cos(x_2 + x_3) + \cdots$$

or anything else. The result is not returned to you in algebraic form, but predicted values and derivatives can be calculated.

The new `npregress` command fits the models using local-linear or local-constant kernel regression. Be aware that fitting accurate nonparametric regressions needs lots of observations. Stata does not limit k, but practical issues do.

You might type

```
. npregress kernel y x1 x2 x3, vce(bootstrap)
```

Reported will be the averages of the partial derivatives of `y` with respect to `x1`, `x2`, and `x3` and their standard errors, which are obtained by bootstrapping. The averages are calculated over the data. After fitting the model, you could obtain predicted values using `predict`.

Average derivatives are something like coefficients, or at least they would be if the model were linear, which it is not. Realize that average derivatives in nonlinear models are not derivatives at the average. You might want to know the derivative of `y` w.r.t. `x1`, `x2`, and `x3` at the average values of `x1`, `x2`, and `x3`. You can use `margins` to obtain that:

```
. margins, dydx(x1 x2 x3) atmeans
```

Or perhaps you want the predicted values evaluated at specific points of interest,

```
. margins, at(x1=2 x2=3 x3=1) at(x1=2 x2=3 x3=2)
```

If you wanted `x3` to be 1, 2, . . . , 10, you could type

```
. margins, at(x1=2 x2=3 x3=1(1)10)
```

Then, you could type

```
. marginsplot
```

to graph this slice of the function.

By the way, `margins` not only makes calculations, it can also produce bootstrap standard errors for them.

See [R] **npregress**.

Highlight 12. Power analysis for linear regression, cluster randomized designs, and your own methods

Stata's `power` command performs power and sample-size analysis (PSS). Its features now include PSS for linear regression and for cluster randomized designs (CRDs). In addition, you can now add your own power and sample-size methods to the `power` command.

The new PSS methods for linear regression include the following:

- `power oneslope` performs PSS for a slope test in a simple linear regression. It computes sample size or power or the target slope given other study parameters. See [PSS] **power oneslope**.
- `power rsquared` performs PSS for an R^2 test in a multiple linear regression. An R^2 test is an F test for the coefficient of determination (R^2). The test can be used to test the significance of all the coefficients, or it can be used to test a subset of them.

  In both cases, `power rsquared` computes sample size or power or the target R^2 given other study parameters. See [PSS] **power rsquared**.
- `power pcorr` performs PSS for a partial-correlation test in a multiple linear regression. A partial-correlation test is an F test of the squared partial multiple correlation coefficient. The command computes sample size or power or the target squared partial correlation coefficient given other study parameters. See [PSS] **power pcorr**.

The new PSS methods for CRDs include the following:

The five existing `power` methods listed below, extended to support CRDs or clustered data when you specify new option `cluster`.

Command	Title
`power onemean, cluster`	One-sample mean test in a CRD
`power oneproportion, cluster`	One-sample proportion test in a CRD
`power twomeans, cluster`	Two-sample means test in a CRD
`power twoproportions, cluster`	Two-sample proportions test in a CRD
`power logrank, cluster`	Log-rank test in a CRD

In a CRD, groups of subjects (clusters) are randomized instead of individual subjects, so the sample size is determined by the number of clusters and the cluster size. The sample-size determination consists of the determination of the number of clusters given cluster size or

the determination of cluster size given the number of clusters. The commands compute one of the number of clusters, cluster size, power, or minimum detectable effect size given other parameters and provide options to adjust for unequal cluster sizes.

For two-sample methods, you can also adjust for unequal numbers of clusters in the two groups.

As with all other power methods, the new methods allow you to specify multiple values of parameters and automatically produce tabular and graphical results.

The final new feature is that you can add your own PSS methods. How you do that is now documented in [PSS] ***power usermethod***. It is easy to do. You write a program that computes sample size or power or effect size. The `power` command will do the rest for you. It will deal with the support of multiple values in options and with the automatic generation of graphs and tables of results.

Highlight 13. Produce PDF and Word documents

It is now just as easy to produce PDF and Word documents in Stata as it is to produce Excel worksheets. Everybody loved `putexcel` in Stata 14. If you are among them, you will love `putpdf` and `putdocx`.

They work just like `putexcel`. That means you can write do-files to create entire PDF or Word reports containing the latest results, tables, and graphs. You can automate reproducible reports.

The new `putpdf` command writes paragraphs, images, and tables to a PDF file. Images include Stata graphs and other images such as your organization's logo. You can format the objects, too—bold face, italics, size, custom tables, etc.

The new `putdocx` command writes paragraphs, images, and tables to a Word file or, to be precise about it, to Office Open XML (`.docx`) files. Just as with `putpdf`, images include Stata graphs, and you can format the objects.

See [P] **putpdf** and [P] **putdocx**.

Highlight 14. Graph color transparency or opacity

Up until now, graph one thing on top of any other, and the object on top covered up the object below. In the jargon, Stata's colors are fully opaque or, if you prefer, not at all transparent. Stata 15 still works that way by default. Stata 15, however, allows you to control the opacity (transparency) of its colors.

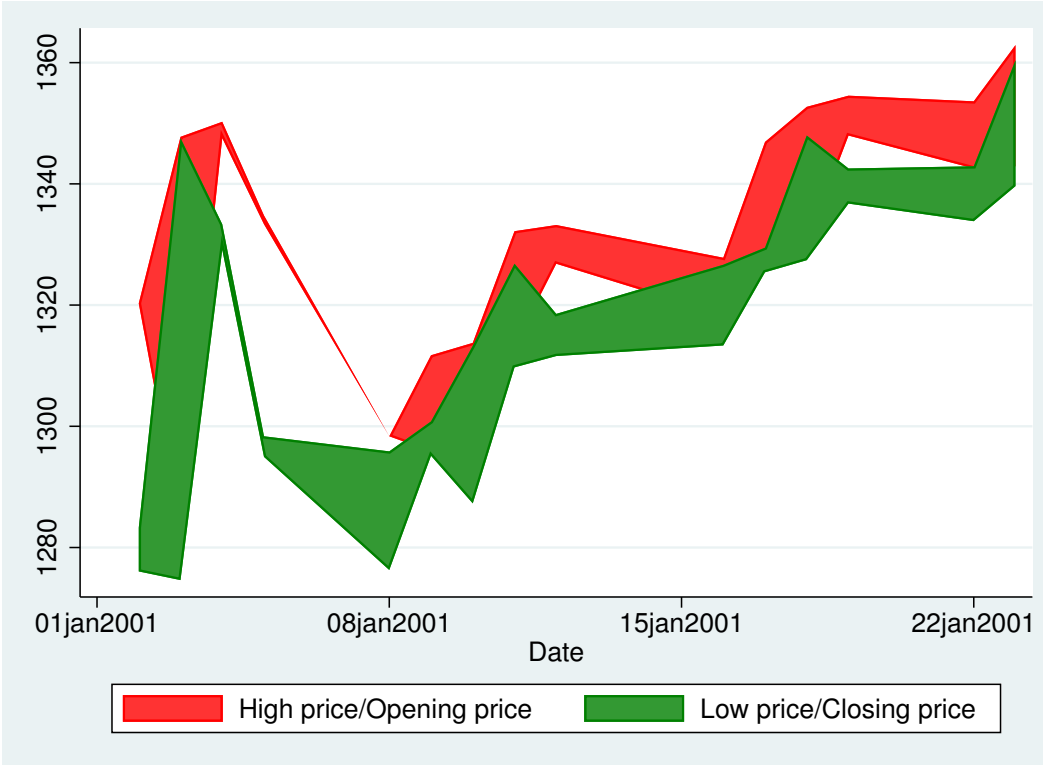
Opacity is specified at a percent. By default, Stata's colors are 100% opaque.

You can specify opacity whenever you specify a color, such as in the `mcolor()` option, which controls the colors of markers. Rather than specifying `green`, you can now specify `green%50`. Rather than specifying `"0 255 0"` (equivalent to green), you can specify `"0 255 0%50"`. And you can simply specify `%50` to make the default color 50% opaque.

You usually do not want to specify `%0`. Yes, it is fully transparent, but it is also invisible.

Here is a graph where we use %70:

```
. twoway rarea high open date in 1/15, color(red%70)   ||
>        rarea low close date in 1/15, color(green%70)
```



Highlight 15. ICD-10-CM and ICD-10-PCS support

Stata 15 now supports ICD-10-CM and ICD-10-PCS, the U.S. ICD-10 codes provided by the National Center for Health Statistics (NCHS) and the Centers for Medicare and Medicaid Services (CMS). These are the codes mandated for all medical billing in the United States.

Stata has long supported ICD codes for reporting medical diagnosis, procedures, and mortality. ICD stands for international statistical classification of diseases and related health problems. Stata began support of the code in 1998 starting with ICD-9-CM version 16 and supported all revisions after that.

Stata supports ICD-10 codes revisions since 2003.

See [D] **icd**, [D] **icd10cm**, and [D] **icd10pcs**.

Highlight 16. Federal Reserve Economic Data support

The St. Louis Federal Reserve makes available over 470,000 U.S. and international economic and financial time series to registered users. Registering is free and easy to do. The service is called FRED. FRED includes data from 84 sources, including the Federal Reserve, the Penn World Table, Eurostat, and the World Bank.

In Stata 15, you can use Stata's GUI to access and download FRED data. You search or browse by category or release or source. You click to select series of interest. Select 1 or select 100. When you click on **Import**, Stata will download them and combine them into a single, custom dataset in memory.

These same features are also available from Stata's command line interface. The command is `import fred`. The command is convenient when you want to automate updating the 27 different series that you are tracking for a monthly report.

Stata can access FRED and it can access ALFRED. ALFRED is FRED's historical archive data.

See [D] **import fred**.

1.3.2 What's new in statistics (general)

Finite mixture models is *Highlight 9* of the release, mixed-logit models is *Highlight 10*, and nonparametric regression is *Highlight 11*.

Also new are the following:

1. **bayes: prefix works with general-purpose estimation commands**

The new `bayes:` prefix (*Highlight 2* of the releases) works with many of the general-purpose estimation commands:

Command	Purpose
betareg	Beta regression
binreg	Binomial regression
biprobit	Bivariate probit regression
clogit	Conditional logistic regression
cloglog	Complementary log-log regression
fracreg	Fractional-outcome regression
glm	Generalized linear model
gnbreg	Generalized negative binomial regression
heckman	Heckman selection model
heckoprobit	Heckman ordered probit with sample selection
heckprobit	Heckman probit with sample selection
hetprobit	Heteroskedastic probit regression
hetregress	Heteroskedastic linear regression
intreg	Interval linear regression
logistic	Logistic regression (default odds ratio)
logit	Logistic regression (default coefficients)
mlogit	Multinomial logistic regression
mprobit	Multinomial probit regression
mvreg	Multivariate linear regression
nbreg	Negative binomial regression
ologit	Ordered logistic regression
oprobit	Ordered probit regression
poisson	Ordered Poisson regression
probit	Probit regression
regress	Linear regression
tnbreg	Truncated negative binomial regression
tobit	Tobit regression
tpoisson	Truncated Poisson regression
truncreg	Truncated linear regression
zinb	Zero-inflated negative binomial regression
zioprobit	Zero-inflated ordered probit regression
zip	Zero-inflated Poisson regression

The list above is of general-purpose estimation commands. The `bayes:` prefix works with multilevel estimation commands, too.

See [BAYES] **bayes** and [BAYES] **bayesian estimation**.

2. **New command fits heteroskedastic regression**

   New estimation command `hetregress` fits heteroskedastic regression by modeling the variance as an exponential function of the covariates you specify. Two estimation methods, maximum likelihood and Harvey's two-step generalized least squares, are provided. Robust, cluster–robust, bootstrap, and jackknife standard errors are supported. Survey-data estimation is supported via the `svy` prefix.

   See [R] **hetregress**.

3. **New command fits Poisson regression with Heckman-style selection**

   New estimation command `heckpoisson` fits a Poisson regression model with endogenous sample selection. All the standard postestimation features are provided.

   See [R] **heckpoisson**.

4. **New command fits zero-inflated ordered probit regression**

   New estimation command `zioprobit` fits zero-inflated ordered probit models. This model is used when data exhibit a higher fraction of the zeros than is expected from a standard ordered probit model.

   We say 0, imagining that the dependent variable contains 0, 1, 2, ..., but we mean the lowest value of the outcome variable because it could just as well contain 2, 5, 9, ...

   The zero inflation is accounted for by assuming that the zeros come from both a probit model and an ordered probit model. Each model can have different covariates.

   See [R] **zioprobit**.

5. **Tobit now accepts censoring limits and constraints**

   Some people think of tobit as being censored at zero. Stata's `tobit` estimation command allows you to specify the lower value of the censoring point, and it allows you to specify an upper censoring point, too. All that is unchanged. You can now specify censoring points—upper, lower, or both—that vary observation by observation. The censoring points can be stored in variables.

   `tobit` now allows constraints.

   `tobit` now has the other standard features that it always should have had, but that is just for completeness. You can, for instance, specify initial values.

   See [R] **tobit**.

6. **tpoisson, ul()**

   Existing estimation command `tpoisson` fits truncated Poisson models. It previously fit left-truncated models only. It now fits left-, right-, and both-truncated models. New option `ul()` specifies the upper truncation limit.

   See [R] **tpoisson**.

7. **Factor variables now work more like you would expect they would**

   Consider fitting a model with the terms

   ```
   . est_command ... i.a i(2 3).a#i.b ...
   ```

   What should happen? What happens now should be more in line with your expectations. `i.a` adds main-effect coefficients for each level of `a`, and the interaction `i(2 3).a#i.b` is restricted to `a`'s levels 2 and 3.

   What used to happen was rather more surprising. The entire RHS of the model was restricted to levels 2 and 3 of `a`.

8. **One- and two-sample mean tests with clustered data**

   Existing command `ztest` has new option `cluster()` and other new options to account for clustering.

   See [R] **ztest**.

9. **One- and two-sample proportion tests with clustered data**

   Existing command `prtest` has new option `cluster()` and other new options to account for clustering.

   See [R] **prtest**.

10. **Note explaining interpretation of intercept when exponentiated coefficients reported**

    Many estimation commands report exponentiated coefficients, either by default or because you specified an option requesting the odds ratio, incidence rate ratio, hazard ratio, and so on. In those cases, Stata also reports the exponentiated intercept. This confuses some people, especially students. Stata now adds a note at the end of the output explaining the interpretation of the exponentiated intercept.

    Notes also make clear which parameters are exponentiated.

11. **ivtobit has improved convergence**

    Existing estimation command `ivtobit` fits instrumental-variable tobit models. It now converges more reliably when there are two or more endogenous variables.

12. **New dots( ) option with replication methods**

    Existing prefix commands `bootstrap`, `jackknife`, `permute`, and `simulate` have new option `dots(#)`, which displays dots every # replications. This provides entertainment and confirmation that the command is still working during long runs.

    See [R] **bootstrap**, [R] **jackknife**, [R] **permute**, and [R] **simulate**.

13. **Option noskip renamed lrmodel**

    Existing estimation commands `biprobit`, `heckman`, `heckprobit`, `hetprobit`, and `truncreg` had option `noskip`, which presented the model test as a likelihood-ratio rather than the default Wald test. This option has been renamed `lrmodel`. The old option name continues to work. (There was a justification for the old name. Calculating the likelihood-ratio test requires fitting the constant-only model. `noskip` specified that fitting of that model not be skipped!)

14. **hetprobit, waldhet (option rename)**

    Existing estimation command `hetprobit` fits heteroskedastic probit models. Existing option `nolrtest` has been renamed `waldhet`. It tests whether the variance is heteroskedastic. Old option `nolrtest` continues to work under version control.

    See [R] **hetprobit**.

15. **Stata names free parameters in fitted models differently**

    Free parameters are scalar parameters, variances, covariances, and the like that are part of the model being fit. A free parameter might be $\ln(\sigma)$.

    We have made a deep change in the internals of Stata. What does this mean for you? Not much. If a model fits free parameter `/lnsigma`, you can no longer refer to its value as `_b[lnsigma:_cons]`. You must refer to it as `_b[/lnsigma]`. You probably always referred to it that way. It involved less typing.

The renaming might matter in programs that you write; see the next item.

Old syntax is preserved under version control.

See [R] **ml** and see the next item.

16. **Program your own models? ml uses the new free-parameter syntax**

`ml` now allows and prefers that free parameters be specified as */name*. You can no longer refer to them as if they were constant-only equations, which is to say *name*`:_cons`, except under version control.

As explained in the previous item, `_b[/lnsigma]` is no longer a shorthand for `_b[lnsigma:_cons]`. `_b[/lnsigma]` is its own thing. It is the free parameter named `/lnsigma` and not the constant term from equation `lnsigma`.

If you were using `ml` to fit your own maximum likelihood model, you might create `/lnsigma` thinking you were creating an equation named `lnsigma`. You are not. Now you are creating a single free parameter. If you want to create a constant-only equation, you must use `(lnsigma:)`, which always meant to create an "equation" called `lnsigma`.

There are other implications for programmers writing advanced code. Matrix row and column names have changed and are now easier to use. This is mentioned in *What's new in programming*.

See [R] **ml**.

17. **More stored results**

Regular commands store their results in `r()`, and estimation commands store them in `e()`. There are now more of them. If something is reported, a result is stored even if what is reported could be calculated from the other stored results.

1.3.3 What's new in statistics (multilevel)

Multilevel mixed-effects models are also known as hierarchical models, nested data models, mixed models, random coefficient models, random effects models, random parameter models, and split-plot designs.

Nonlinear mixed models are *Highlight 6* of the release.

Also new are the following:

18. **Multilevel mixed-effects tobit regression**

New estimation command `metobit` fits tobit models. In tobit models, outcomes below a limit or above a limit are censored. Limits can be fixed (say, 0 and 1,000) or vary observation by observation.

See [ME] **metobit**.

19. **Multilevel mixed-effects interval regression**

New estimation command `meintreg` fits interval regression models. In these models, the exact value of the dependent variable is not observed in some or all observations. Instead, y_i is known to be within $[a_i, b_i]$. Ranges can be open ended, so the model handles censoring as well as intervals.

See [ME] **meintreg**.

20. **bayes: prefix works with multilevel models**

Stata's new `bayes:` prefix command (see *Highlight 2*) may be used with the following:

Estimation command	Fits multilevel mixed-effects ...
`bayes: mixed`	linear regression
`bayes: metobit`	tobit regression
`bayes: meintreg`	interval regression
`bayes: melogit`	logistic regression
`bayes: meprobit`	probit regression
`bayes: mecloglog`	complementary log-log regression
`bayes: mepoisson`	Poisson regression
`bayes: menbreg`	negative binomial regression
`bayes: meglm`	generalized linear model
`bayes: mestreg`	parametric survival models

See [BAYES] **bayes**.

21. **Standard deviations and correlations instead of variances and covariances**

New postestimation command `estat sd` displays random effects and within-group error parameter estimates as standard deviations and correlations instead of the variances and covariances reported in the estimation output.

See [ME] **estat sd**.

1.3.4 What's new in statistics (Bayesian)

The new `bayes:` prefix command is *Highlight 2* of the release.

Also new are the following:

22. **bayesmh, eform( )**

Existing estimation command `bayesmh` now allows the `eform` and `eform(`*string*`)` options for reporting exponentiated coefficients such as odds ratios, incidence rate ratios, and the like.

See [BAYES] **bayesmh**.

23. **bayesmh, show( )**

Existing estimation command `bayesmh` now allows new option `show(`*paramlist*`)` to specify which model parameters should be presented in the output. Option `show()` joins existing option `noshow()`. Specify one, the other, or neither.

See [BAYES] **bayesmh**.

24. **bayesmh, showreffects**

Existing estimation command `bayesmh` now allows new option `showreffects` to specify that all random-effects estimates be presented in the output. They are not displayed by default.

See [BAYES] **bayesmh**.

25. **Postestimation supports new bayes: prefix command**

If you use the new `bayes:` prefix command with multilevel models such as `mixed` or `meglm`, `bayesgraph`, `bayesstats ess`, and `bayesstats summary` have new options.

New option `showreffects` displays the results for all random-effects parameters.

New option `showreffects()` displays specified random-effects parameters.

By default, results are displayed for all model parameters except the random-effects parameters.

See [BAYES] **bayesian postestimation**.

26. `bayesmh` with nonlinear models has the following updates to the linear-combination specifications within substitutable expressions:

    1. When you specify `{xb: x1 x2}`, the constant term is included automatically. That is, `{xb: x1 x2}` is equivalent to

       ```
       {xb:x1}*x1 + {xb:x2}*x2 + {xb:_cons}
       ```

       You can suppress the constant term by specifying the new `noconstant` option such as `{xb: x1 x2, noconstant}`.

    2. The new `xb` option is required when you include only one variable in the linear-combination specification such as `{xb: z, xb}`. The specification `{xb:z}` without the `xb` option corresponds to either a free parameter named `z` with the grouping label `xb` or a regression coefficient on variable `z` that was included in the previously defined linear combination `xb`.

    3. Regression coefficients of linear combinations are now defined as {*xbname*`:`*varname*} instead of {*xbname*`_`*varname*}. For example, if you specify a linear combination `{xb: x1 x2}`, you refer to regression coefficients of variables `x1` and `x2` as `{xb:x1}` and `{xb:x2}` instead of `{xb_x1}` and `{xb_x2}`.

    The old behavior of linear-combination specifications is available under version control.

27. **Programmer alerts**

    If you consume matrix results from `e()` after `bayesmh`, be aware of two small labeling issues:

    1. Terms that were marked as omitted were not stored in the matrix's row and column names. They are now. Old behavior is available under version control.

    2. `bayesmh` now includes equation labels in the row and column names of `e(mean)` and `e(median)`.

1.3.5 What's new in statistics (power and sample size)

New is the following:

28. **Power analysis for linear regression, cluster randomized designs, and your own methods**

    Stata's `power` command now includes power and sample-size analysis for linear regression and for cluster randomized designs. In addition, you can now add your own power and sample-size methods to the `power` command.

    This is *Highlight 12* of the release.

1.3.6 What's new in statistics (survival analysis)

Stata's new ability to fit interval-censored parametric survival models is *Highlight 8* of the release.

Also new are the following:

29. **bayes: streg**

    The new `bayes:` prefix command (*Highlight 2*) can be used with existing estimation command `streg` to fit Bayesian parametric survival models.

    See [BAYES] **bayes: streg**.

30. **fmm: streg**

    The new `fmm:` prefix command (*Highlight 9*) can be used with `streg` to fit finite mixtures of parametric survival models. See [FMM] **fmm: streg**.

31. **streg, strata(i.varname)**

    Existing estimation command `streg`'s option `strata()` now allows a factor variable as an argument. You can specify `strata(i.agegroup)` for instance. You can specify `strata(i(2 4 6).agegroup` if you want the stratum to be 2, 4, 6, and treat the other levels as baseline.

    Previously, you specified `strata(agegroup)`, and the option created new dummy variables in the dataset to include them in the model. If you specify `strata(agegroup)`, it is now interpreted as if you specified `strata(i.agegroup)`. Old behavior is preserved under version control.

32. **Better names for streg's free parameters**

    Free parameters in `streg` models now have more descriptive names. The scale parameter $\ln\sigma$ is now named `/lnsigma`, for instance, and not `/ln_sig`. What was named `/ln_gam` is now named `/lngamma`. What was named `/ln_the` is now named `/lntheta`. You use the new names with `_b[]`. You continue to use the old names under version control.

1.3.7 What's new in statistics (survey data)

New features are the following:

33. These new estimation commands may be used with the `svy:` prefix:

Command	Purpose
`svy: asmixlogit`	Alternative-specific mixed logit regression
`svy: heckpoisson`	Poisson regression with sample selection
`svy: hetregress`	Heteroskedastic linear regression
`svy: stintreg`	Parametric interval-censored survival regression
`svy: zioprobit`	Zero-inflated ordered probit
	Multilevel mixed-effects ...
`svy: metobit`	tobit regression
`svy: meintreg`	interval regression
`svy: eregress`	Extended linear regression
`svy: eintreg`	Extended interval regression
`svy: eprobit`	Extended probit regression
`svy: eoprobit`	Extended ordered probit regression
`svy: gsem`	Generalized SEM, including latent class analysis

See [SVY] **svy**.

34. The following existing estimation commands support combined use of `svy:` and `fmm:` to fit survey-adjusted finite mixture models:

Command	Purpose
`svy: fmm: regress`	Linear regression
`svy: fmm: tobit`	Tobit regression
`svy: fmm: intreg`	Interval regression
`svy: fmm: truncreg`	Truncated regression
`svy: fmm: ivregress`	Instrumental-variable regression
`svy: fmm: logit`	Logistic regression
`svy: fmm: probit`	Probit regression
`svy: fmm: cloglog`	Conditional log-log regression
`svy: fmm: ologit`	Ordered logistic regression
`svy: fmm: oprobit`	Ordered probit regression
`svy: fmm: mlogit`	Multinomial logistic regression
`svy: fmm: poisson`	Poisson regression
`svy: fmm: nbreg`	Negative binomial regression
`svy: fmm: tpoisson`	Truncated Poisson regression
`svy: fmm: betareg`	Beta regression
`svy: fmm: glm`	Generalized linear model
`svy: fmm: streg`	Parametric survival regression

See [SVY] **svy** and [FMM] **fmm**.

35. **New dots( ) option**

Existing prefix commands `svy bootstrap:`, `svy jackknife:`, `svy brr:`, and `svy sdr:` allow new option `dots(#)`. It displays a dot every # replications.

See [SVY] **svy bootstrap**, [SVY] **svy jackknife**, [SVY] **svy brr**, and [SVY] **svy sdr**.

1.3.8 What's new in statistics (SEM)

Stata 15's new latent class analysis capabilities are *Highlight 1* of the release. Existing estimation command `gsem` performs latent class analysis.

Also new are the following:

36. **Likelihood ratio, AIC, and BIC after classic latent class analysis**

New postestimation command `estat lcgof` reports the G^2 likelihood-ratio test after fitting classic latent class models (logit outcome models). The test compares the fitted model to the saturated model. The new command also reports AIC and BIC.

See [SEM] **estat lcgof**.

37. **Marginal means across latent classes**

New postestimation command `estat lcmean` computes marginal means in latent class models for all response variables across the latent classes.

See [SEM] **estat lcmean**.

38. **Predicted marginal probability of class membership**

    New postestimation command `estat lcprob` reports the marginal probabilities of class membership in latent class models.

    See [SEM] **estat lcprob**.

39. **New predictions after fitting latent class models**

    Existing postestimation command `predict` has new options after fitting latent class models. They are

    - `predict, classpr` to predict latent class probabilities;
    - `predict, classposteriorpr` to predict posterior latent class probabilities;
    - `predict, mu marginal` to predict the overall expected value of each outcome by summing the latent class means weighted by the latent class probabilities;
    - `predict, mu pmarginal` to predict the overall expected value of each outcome by summing the latent class means weighted by the posterior latent class probabilities; and
    - `predict, mu class(#)` to predict the expected value of each outcome for class #.

    See [SEM] **predict after gsem**.

40. **gsem now fits multiple-group models**

    Existing estimation command `gsem`, whether used to fit the new LCA models or the existing generalized SEM models, now allows the `group()` option just as command `sem` does. You can type

    ```
    . gsem ..., group(agegrp)
    ```

41. **sem and gsem report multiple-group models in separate tables**

    Both `sem` and `gsem` now report models in a more readable format. Rather than a single table encompassing all multiple group parameters, separate tables are produced.

    New option `byparm` produces the old output.

    See [SEM] **sem reporting options** and [SEM] **gsem reporting options**.

42. **gsem now fits truncated Poisson models**

    `gsem`, whether used to fit the new LCA models or the existing generalized SEM models, now fits truncated Poisson models if you specify option `family(poisson, ltruncated(...))`.

    See [SEM] **gsem family-and-link options**.

43. **Variances and covariances as standard deviations and correlations**

    New postestimation command `estat sd` after `gsem` reports the estimated variance components as standard deviations and correlations.

    See [SEM] **estat sd**.

1.3.9 What's new in statistics (panel data)

New features are the following:

44. **Cointegration test for nonstationary process in panel data**

    The new `xtcointtest` command tests for cointegration in nonstationary panel data. It provides three methods, the ones due to Kao, Pedroni, and Westerlund. All assume the same null hypothesis but differ on their specification of the alternative hypotheses.

    See [XT] **xtcointtest**.

45. **Option noskip renamed lrmodel**

    Existing estimation commands `xtcloglog`, `xtintreg`, `xtlogit`, `xtnbreg`, `xtologit`, `xtoprobit`, `xtpoisson`, `xtprobit`, `xtstreg`, and `xttobit` had option `noskip`, which presented the model test as a likelihood-ratio rather than the default Wald test. This option has been renamed `lrmodel`. The old option name continues to work. (There was a justification for the old name. Calculating the likelihood-ratio test requires fitting the constant-only model. `noskip` specified that fitting of that model not be skipped!)

1.3.10 What's new in statistics (time series)

Stata 15's new support for retrieving Federal Reserve Economic Data (FRED) is *Highlight 16* of the release.

Also new are the following:

46. **Threshold regression**

    New estimation command `threshold` fits threshold regressions. These are linear regressions in which the coefficients change by estimated cutpoints. This could be on the basis of time. Then you have one set of coefficients before the first threshold, another set after the first and before the second, and so on.

    Or it could be on the basis of an exogenous variable. In that case, you would have a set of coefficients when `x` $<$ the first threshold, another set after the first and before the second, and so on.

    The lagged value of the dependent variable is an example of an exogenous variable. In that case, you would have a set of coefficients when `l.y` $<$ the first threshold, another set after the first and before the second, and so on. This last case is known as the self-exciting threshold model.

    You can specify or estimate the number of threshold points.

    See [TS] **threshold**.

47. **Test for structural breaks after time-series regression**

    The new `estat sbcusum` command is for use after `regress` on `tsset` data.

    It tests for stability of coefficients based on the cumulative sum (cusum) of either the recursive residuals or the ordinary least-squares residuals. This can be used to test for structural breaks due to changes in regression coefficients over time.

    `estat sbcusum` also plots the cusum versus time along with confidence bands. The graph provides additional information that can help identify time periods in which the coefficients are unstable.

    See [TS] **estat sbcusum**.

48. **rolling, dots( )**

Existing estimation command `rolling` fits rolling window and recursive linear regressions. This can be time consuming. It has new option `dots(#)`, which displays a dot every # replications. This is not only entertaining; it provides information about the percent of the calculation completed.

See [TS] **rolling**.

1.3.11 What's new in statistics (multivariate)

Latent class analysis is *Highlight 1* of the release. It is an alternative to cluster analysis.

Also new is the following:

49. **New bayes: mvreg command**

Stata 15's new `bayes:` prefix command (*Highlight 2* of the release) can be used with existing estimation command `mvreg` to fit Bayesian multivariate regression models.

See [BAYES] **bayes**, [BAYES] **bayesian estimation**, and [MV] **mvreg**.

1.3.12 What's new in functions

Functions are used in expressions. For instance, `log()` is a function:

```
. generate lincome = log(income)
```

The functions listed below are also available in both Stata and Mata.

50. **Cauchy distribution**

A new family of Cauchy distribution functions are provided:

`cauchyden(`a`,`b`,`x`)` computes the density of the Cauchy distribution with location parameter a and scale parameter b.

`cauchy(`a`,`b`,`x`)` computes the cumulative distribution function of the Cauchy distribution with location parameter a and scale parameter b.

`cauchytail(`a`,`b`,`x`)` computes the reverse cumulative Cauchy distribution with location parameter a and scale parameter b.

`invcauchy(`a`,`b`,`p`)` computes the inverse cumulative Cauchy distribution. If `cauchy(`a`,`b`,`x`)` $= p$, then `invcauchy(`a`,`b`,`p`)` $= x$.

`invcauchytail(`a`,`b`,`p`)` computes the inverse reverse cumulative Cauchy distribution. If `cauchytail(`a`,`b`,`x`)` $= p$, then `invcauchytail(`a`,`b`,`p`)` $= x$.

`lncauchyden(`a`,`b`,`x`)` computes the natural logarithm of the density of the Cauchy distribution with location parameter a and scale parameter b.

`rcauchy(`a`,`b`)` is a Cauchy random-number generator. It computes Cauchy random variates with location parameter a and scale parameter b.

See [FN] **Statistical functions** and [FN] **Random-number functions**.

51. **Laplace distribution**

A new family of Laplace distribution functions are provided:

`laplaceden(`m`,`b`,`x`)` computes the density of the Laplace distribution with mean m and scale parameter b.

`laplace(`m`,`b`,`x`)` computes the cumulative distribution function of the Laplace distribution with mean m and scale parameter b.

`laplacetail(`m`,`b`,`x`)` computes the reverse cumulative Laplace distribution with mean m and scale parameter b.

`invlaplace(`m`,`b`,`p`)` computes the inverse cumulative Laplace distribution. If `laplace(`m`,`b`,`x`)` $= p$, then `invlaplace(`m`,`b`,`p`)` $= x$.

`invlaplacetailb(`m`,`b`,`p`)` computes the inverse reverse cumulative Laplace distribution. If `laplacetail(`m`,`b`,`x`)` $= p$, then `invlaplacetail(`m`,`b`,`p`)` $= x$.

`lnlaplaceden(`m`,`b`,`x`)` computes the natural logarithm of the density of the Laplace distribution with mean m and scale parameter b.

`rlaplace(`m`,`b`)` is a Laplace random-number generator. It computes Laplace random variates with mean m and scale parameter b.

See [FN] **Statistical functions** and [FN] **Random-number functions**.

52. **Stream random numbers**

All of Stata's and Mata's existing random-number functions can now produce stream random numbers. Streams are necessary for running simulations and bootstraps simultaneously. Stata's functions previously produced single streams. In a single stream, setting the seed determined the random numbers that would be produced. If two routines running simultaneously set the same seed, they would obtain the same random numbers. Multiple streams let them produce different random numbers. Moreover, stream random-number generators (RNGs) are designed so that you can simultaneously draw random numbers and know that you are drawing from different sequences.

By default, Stata's and Mata's random-number functions are based on an underlying RNG. They are `kiss32` and `mt64`. `mt64` is the default, and `kiss32` was provided for backward compatibility. Now there is a third RNG: `mt64s`, which is the stream version of the Mersenne Twister.

To use stream random numbers, you must first set the RNG to `mt64s`:

```
. set rng mt64s
```

After that, you set the seed the usual way,

```
. set seed #
```

A new command allows you to set the stream,

```
. set rngstream #
```

where $1 \leq \# \leq 32{,}767$. Each stream can produce 2^{128} pseudorandom numbers before the sequence repeats.

Thus you can launch multiple Statas and run the same do-file to produce simulations. Each do-file can (and should) use the same seed. Before starting the do-file, set the `rngstream`, or have the do-file accept a stream argument to set the stream. Or launch the separate Statas in batch mode and specify new start-up option `rngstream#`.

See [R] **set rngstream**.

1.3.13 What's new in graphics

Stata's new features allowing you to specify the transparency or opacity of colors is *Highlight 14* of the release.

Also new are the following:

53. **Scalable vector graphics**

Stata now supports scalable vector graphics, also known as SVGs. Vector graphic format is better than raster format because it is, well, scalable. If you magnify the graph, it does not become grainy or pixelated.

Scalable vector graphics are written in `.svg` files. This format is especially popular for use on web pages.

Use `graph export, as(svg)`.

See [G-2] **graph export**.

54. **New marker symbols**

Markers are used to show where the data lie. Dots, hollow, or solid circles are popular. Stata has lots of marker symbols. Now it has more with short names `X`, `x`, `A`, `a`, `V`, `v`, and `|`. Here are all of Stata 15's marker symbols:

symbolstyle	Synonym (if any)	Description
`circle`	`O`	solid
`diamond`	`D`	solid
`triangle`	`T`	solid
`square`	`S`	solid
`plus`	`+`	
`X`	`X`	
`arrowf`	`A`	filled arrow head
`arrow`	`a`	
`pipe`	`|`	
`V`	`V`	
`smcircle`	`o`	solid
`smdiamond`	`d`	solid
`smsquare`	`s`	solid
`smtriangle`	`t`	solid
`smplus`		
`smx`	`x`	
`smv`	`v`	
`circle_hollow`	`Oh`	hollow
`diamond_hollow`	`Dh`	hollow
`triangle_hollow`	`Th`	hollow
`square_hollow`	`Sh`	hollow
`smcircle_hollow`	`oh`	hollow
`smdiamond_hollow`	`dh`	hollow
`smtriangle_hollow`	`th`	hollow
`smsquare_hollow`	`sh`	hollow
`point`	`p`	a small dot
`none`	`i`	a symbol that is invisible

You cannot rotate the arrows yet, but that is forthcoming.

See [G-4] ***symbolstyle***.

55. **New graph command for use after fitting nonparametric regression models**

New postestimation command `npgraph` is for use after fitting a nonparametric regression model using the new `npregress` command. `npregress` is *Highlight 11* of the release. `npgraph` plots the nonparametric function fit by `npregress` along with a scatterplot of the data.

See [R] **npregress postestimation**.

56. **.gph file format updated**

    Stata's `.gph` file format was updated because of the new transparent colors and marker symbols. Previous Statas will not be able to read the new format, but Stata 15 can read old formats without difficulty.

1.3.14 What's new in data management

Stata's new ICD-10 features is *Highlight 15* of the release, and Stata's new support of Federal Reserve Data Economic Data is *Highlight 16*.

Also new are the following:

57. **use ... in faster**

    Stata's `use` command is now significantly faster when you specify `in` *range*.

58. **Import and export of dBase files**

    New command `import dbase` imports dBase version III and version IV `.dbf` files. dBase was one of the first micro database management systems for microcomputers and is still used today.

    New command `export dbase` exports to dBase IV format.

    See [D] **import dbase**.

59. **Stata/MP allows up to 120,000 variables**

    Stata/MP's increase to 120,000 variables is up from 32,767 variables. Stata/SE continues to support up to 32,767 variables, and Stata/IC continues to support up to 2,047 variables.

60. **statsby, dots( )**

    Existing command `statsby` has a new option `dots(#)` that displays dots every # replications. This provides entertainment and confirmation that the command is still working during long runs.

    See [D] **statsby**.

1.3.15 What's new in programming

Dynamic documents using new command `markdown` is *Highlight 5* of the release. Producing PDF and Word documents using new commands `putpdf` and `putdocx` is *Highlight 13*.

Also new are the following:

61. **New Java plugin features**

    Stata's Java plugins now have features to store and access

    - Stata's returned results
    - Stata's dataset characteristics
    - Stata's `strL` variables as a buffered array
    - Stata's string scalars
    - Stata's variable types
    - Stata's matrices (they could already handle Mata's matrices)

In addition:

1. It is now easier to access Stata's and Mata's matrix elements.
2. Java plugins can now call Stata commands.
3. Java plugins now use a custom class loader.
    a. Stata no longer needs to be restarted after installation of a new Java plugin, and you can now detach plugins without restarting Stata.
    b. The loader allows for isolation of dependencies between plugins.

See [P] **java** and [P] **javacall**.

62. **postfile bug fix**

`postfile` previously allowed variable names sometimes to be reserved words. You could create a variable named `int`, for instance. This bug is fixed under version control.

63. **New features to support new syntax for free parameters**

Stata has new syntax for free parameters in fitted models. We mentioned this from the user's perspective in *What's new in statistics (general)*. To rehash, `/`*name* is no longer a synonym for *name*`:_cons`; `/`*name* is its own thing for free parameters.

Stata has new functions for dealing with matrix row and column names that include `/`*name*:

`coleqnumb(`M`,`s`)` returns the equation number of matrix M associated with column equation s.

`roweqnumb(`M`,`s`)` returns the equation number of matrix M associated with row equation s.

`colnfreeparms(`M`)` returns the number of free parameters in columns of matrix M.

`rownfreeparms(`M`)` returns the number of free parameters in the rows of matrix M.

Stata also has new macro functions:

`local` *lname* `: colnumb` *matrixname string*

`local` *lname* `: rownumb` *matrixname string*

`local` *lname* `: coleqnumb` *matrixname string*

`local` *lname* `: roweqnumb` *matrixname string*

`local` *lname* `: colnfreeparms` *matrixname*

`local` *lname* `: rownfreeparms` *matrixname*

`local` *lname* `: colnlfs` *matrixname*

`local` *lname* `: rownlfs` *matrixname*

`local` *lname* `: colsof` *matrixname*

`local` *lname* `: rowsof` *matrixname*

`local` *lname* `: colvarlist` *matrixname*

`local` *lname* `: rowvarlist` *matrixname*

`local` *lname* `: collfnames` *matrixname*

`local` *lname* `: rowlfnames` *matrixname*

See [FN] **Matrix functions** and [P] **macro**.

1.3.16 What's new in Mata

New are the following:

64. **Cauchy and Laplace distribution functions**

    The Cauchy and Laplace distribution functions added to Stata have been added to Mata, too.

    See *What's new in functions*.

65. **Functions for calculating values and derivatives of the multivariate normal distribution**

    These functions allow fixed or varying limits, means, and variances/covariances/correlations. Define

    L is the vector lower limits (default $-\infty$)

    U is the vector upper limits (default ∞)

    m is the mean vector (default 0)

    R is the correlation matrix

    V is the variance matrix

    The new functions return the value of the multivarite normal distribution between *L* and *U*:

    `mvnormal(`*U*`,` *R*`)`

    `mvnormal(`*L*`,` *U*`,` *R*`)`

    `mvnormalcv(`*L*`,` *U*`,` *M*`,` *V*`)`

    `mvnormalderiv(`*U*`,` *R*`,` *dU*`,` *dR*`)` returns derivatives in *dU*, *dR*

    `mvnormalderiv(`*L*`,` *U*`,` *R*`,` *dL*`,` *dU*`,` *dR*`)` returns derivatives in *dL*, *dU*, *dR*

    `mvnormalcvderiv(`*L*`,` *U*`,` *M*`,` *V*`,` *dL*`,` *dU*`,` *dM*`,` *dV*`)` returns derivatives in *dL*, *dU*, *dM*, *dV*

    There are also versions of the above functions that allow specification of the number of quadrature points.

    See [M-5] **mvnormal()**.

66. **Open Office XML files**

    New functions were added to the suite for generating Open Office XML (`.docx`) files:

    `_docx_append()`

    `_docx_cell_set_span()`

    See [M-5] **_docx*()**.

67. **PDF files**

    New functions were added to the suite for generating PDF files:

    `PdfDocument.setLandscape()`

    `PdfParagraph.addLineBreak()`

    `PdfParagraph.setVAlignment()`

    `PdfTable.setCellBorderWidth()`

    `PdfTable.setCellBorderColor()`

    `PdfTable.setCellMargin()`

    `PdfTable.setRowSplit()`

`PdfTable.addRow()`

`PdfTable.delRow()`

`PdfTable.addColumn()`

`PdfTable.delColumn()`

The following existing functions now have optional arguments and added capabilities:

`PdfTable.setBorderWidth()`

`PdfTable.setBorderColor()`

`PdfTable.fillStataMatrix()`

`PdfTable.fillMataMatrix()`

See [M-5] **Pdf*( )**.

68. **Percent encoding for URLs**

New function `urlencode(`*s* [`,` *useplus*]`)` returns *s* with any reserved characters changed to percent-encoded ASCII. Special characters are replaced by `%` followed by two hexadecimal digits. For instance, each space is replaced with `%20`. If `useplus` is specified and nonzero, spaces are changed to `+`.

New function `urldecode(`*s*`)` returns *s* with percent encoding undone.

See [M-5] **urlencode( )**.

69. **New option moptimize_init_eq_freeparm()**

New function `moptimize_init_eq_freeparm(`*M*`,` *i*`,` {`"on"`|`"off"`}`) specifies whether the equation for the *i*th parameter is to be treated as a free parameter. This setting is ignored if there are independent variables or an offset attached to the parameter. Free parameters have a shortcut notation that distinguishes them from constant linear equations. The free parameter notation for an equation labeled name is `/`*name*. The corresponding notation for a constant linear equation is *name*`:_cons`.

See [M-5] **moptimize( )**.

1.3.17 What's new in the interface

New are the following:

70. **Do-file Editor is improved**

The Do-file Editor is improved. In Stata for Windows,

- the current (active) line is now highlighted;
- colors for line numbers, margins, bookmarks, etc. can be set;
- bookmarks can be added or deleted by clicking in the bookmarks margin; and
- high-contrast mode is better supported.

Under all operating systems, including Windows,

- column-mode selection and editing are now provided;
- the new indentation guide aids in writing clean code by displaying vertical lines at every tab stop;
- character encoding of legacy do-files can now be specified so that any extended ASCII characters are converted to the right Unicode character;

- comments (`/**/` and `//`) can be added or removed from a selection;
- code folding for `program`, `mata`, and `input` is now provided;
- wrapped lines can be marked visually;
- program code can be automatically reindented to be properly aligned, and spaces are converted to tabs.

Concerning column-mode editing: use *Alt*+mouse dragging or *Alt*+*Shift*+arrow keys on Windows. Substitute *Option* for *Alt* on Mac and *Ctrl* for *Alt* on Linux.

71. **set more off now the default**

Stata displays —`more`— when output is about to scroll off the screen. You press the space bar or click on the **More** button, and another page of output appears. This is called `set more on`.

`set more off` is now the default.

If you prefer the old behavior, type `set more on, permanently`.

See [R] **more**.

72. **If you do set more on** . . .

The **More** button has a useful new feature. When output is paused, click on the triangle to the right of the **More** button. You will have two choices:

Show more results

Run to completion

Click on **Run to completion**, and `more` will be temporarily turned off until the currently running command (or do-file!) completes.

See [R] **more**.

73. **Option to suppress header and footer in logs**

`log` has new option `nomsg`, which suppresses placing the header and footer in the log. The header reports the filename and date and time that the log was opened, and the footer reports the filename and date and time that the log was closed.

See [R] **log**.

74. **Swedish language support**

Swedish now joins Spanish and Japanese as languages for which Stata's menus, dialogs, and the like can be displayed. Manuals and help files remain in English.

If your computer locale is set to Sweden, Stata will automatically use its Swedish setting. To change languages manually, select **Edit** > **Preferences** > **User-interface language...** using Windows or Unix, or on the Mac, select **Stata 15** > **Preferences** > **User-interface language...**. You can also change the language from the command line; see [P] **set locale_ui**.

StataCorp gratefully acknowledges the efforts of Metrika Consulting AB, Stata's official distributor in Sweden, Finland, Norway, and Denmark, for the translation to Swedish.

1.3.18 What's more

We have not listed all the changes, but we have listed the important ones.

Stata is continually being updated. Those between-release updates are available for free over the Internet.

Type `update query` and follow the instructions.

We hope that you enjoy Stata 15.

1.4 References

Chambers, J. M., W. S. Cleveland, B. Kleiner, and P. A. Tukey. 1983. *Graphical Methods for Data Analysis*. Belmont, CA: Wadsworth.

Everitt, B. S., and A. Skrondal. 2010. *The Cambridge Dictionary of Statistics*. 4th ed. Cambridge: Cambridge University Press.

Gould, W. W. 2014. Putting the Stata Manuals on your iPad. *The Stata Blog: Not Elsewhere Classified*. http://blog.stata.com/2014/10/28/putting-the-stata-manuals-on-your-ipad/.

Heyde, C. C., and E. Seneta, ed. 2001. *Statisticians of the Centuries*. New York: Springer.

Johnson, N. L., and S. Kotz, ed. 1997. *Leading Personalities in Statistical Sciences: From the Seventeenth Century to the Present*. New York: Wiley.

Upton, G. J. G., and I. T. Cook. 2014. *A Dictionary of Statistics*. 3rd ed. Oxford: Oxford University Press.