# The Stata Journal

The *Stata Journal* publishes reviewed papers together with shorter notes or comments, regular columns, book reviews, and other material of interest to Stata users. Examples of the types of papers include 1) expository papers that link the use of Stata commands or programs to associated principles, such as those that will serve as tutorials for users first encountering a new field of statistics or a major new technique; 2) papers that go "beyond the Stata manual" in explaining key features or uses of Stata that are of interest to intermediate or advanced users of Stata; 3) papers that discuss new commands or Stata programs of interest either to a wide spectrum of users (e.g., in data management or graphics) or to some large segment of Stata users (e.g., in survey statistics, survival analysis, panel analysis, or limited dependent variable modeling); 4) papers analyzing the statistical properties of new or existing estimators and tests in Stata; 5) papers that could be of interest or usefulness to researchers, especially in fields that are of practical importance but are not often included in texts or other journals, such as the use of Stata in managing datasets, especially large datasets, with advice from hard-won experience; and 6) papers of interest to those who teach, including Stata with topics such as extended examples of techniques and interpretation of results, simulations of statistical concepts, and overviews of subject areas.

For more information on the *Stata Journal*, including information for authors, see the web page

http://www.stata-journal.com

The *Stata Journal* is indexed and abstracted in the following:

- Science Citation Index Expanded (also known as SciSearch®)
- CompuMath Citation Index®



The *Stata Journal*, electronic version (ISSN 1536-8734) is a publication of Stata Press. Stata and Mata are registered trademarks of StataCorp LP.

The Stata Journal (2008)
**8**, Number 1, pp. 139–141

# Stata tip 58: nl is not just for nonlinear models

Brian P. Poi
StataCorp
College Station, TX
bpoi@stata.com

## 1 Introduction

The `nl` command makes performing nonlinear least-squares estimation almost as easy as performing linear regression. In this tip, three examples are given where `nl` is preferable to `regress`, even when the model is linear in the parameters.

## 2 Transforming independent variables

Using the venerable `auto` dataset, suppose we want to predict the weight of a car based on its fuel economy measured in miles per gallon. We first plot the data:

```
. sysuse auto
. scatter weight mpg
```

Clearly, there is a negative relationship between `weight` and `mpg`, but is that relationship linear? The engineer in each of us believes that the amount of gasoline used to go one mile should be a better predictor of weight than the number of miles a car can go on one gallon of gas, so we should focus on the reciprocal of `mpg`. One way to proceed would be to create a new variable, `gpm`, measuring gallons of gasoline per mile and then to use `regress` to fit a model of `weight` on `gpm`. However, consider using `nl` instead:

```
. nl (weight = {b0} + {b1}/mpg)
(obs = 74)
Iteration 0:  residual SS =  1.19e+07
Iteration 1:  residual SS =  1.19e+07
      Source |       SS       df       MS
-------------+------------------------------
       Model |  32190898.6     1  32190898.6         R-squared     =    0.7300
    Residual |  11903279.8    72   165323.33         Adj R-squared =    0.7263
-------------+------------------------------         Root MSE      =  406.5997
       Total |  44094178.4    73  604029.841         Res. dev.     =  1097.134

------------------------------------------------------------------------------
      weight |      Coef.   Std. Err.      t    P>|t|     [95% Conf. Interval]
-------------+----------------------------------------------------------------
         /b0 |   415.1925   192.5243     2.16   0.034     31.40241    798.9826
         /b1 |   51885.27   3718.301    13.95   0.000     44472.97    59297.56
------------------------------------------------------------------------------
  Parameter b0 taken as constant term in model & ANOVA table
```

 st0141

(You can verify that $R^2$ from this model is higher than that from a linear model of `weight` on `mpg`. You can also verify that our results match those from `regress`ing `weight` on `gpm`.)

Here a key advantage of `nl` is that we do not need to create a new variable containing the reciprocal of `mpg`. When doing exploratory data analysis, we might want to consider using the natural log or square root of a variable as a regressor, and using `nl` saves us some typing in these cases. In general, instead of typing

```
. generate sqrtx = sqrt(x)
. regress y sqrtx
```

we can type

```
. nl (y = {b0} + {b1}*sqrt(x))
```

# 3 Marginal effects and elasticities

Using `nl` has other advantages as well. In many applications, we include not just the variable $x$ in our model but also $x^2$. For example, most wage equations express log wages as a function of experience and experience squared. Say we want to fit the model

$$y_i = \alpha + \beta_1 x_i + \beta_2 x_i^2 + \epsilon_i$$

and then determine the elasticity of $y$ with respect to $x$; that is, we want to know the percent by which $y$ will change if $x$ changes by one percent.

Given the interest in an elasticity, the inclination might be to use the `mfx` command with the `eyex` option. We might type

```
. generate xsq = x^2
. regress y x xsq
. mfx compute, eyex
```

These commands will not give us the answer we expect because `regress` and `mfx` have no way of knowing that `xsq` is the square of `x`. Those commands just see two independent variables, and `mfx` will return two "elasticities", one for `x` and one for `xsq`. If $x$ changes by some amount, then clearly $x^2$ will change as well; however, `mfx`, when computing the derivative of the regression function with respect to `x`, holds `xsq` fixed!

The easiest way to proceed is to use `nl` instead of `regress`:

```
. nl (y = {a} + {b1}*x + {b2}*x^2), variables(x)
. mfx compute, eyex
```

Whenever you intend to use `mfx` after `nl`, you must use the `variables()` option. This option causes `nl` to save those variable names among its estimation results.

# 4 Constraints

`nl` makes imposing nonlinear constraints easy. Say you have the linear regression model

$$y_i = \alpha + \beta_1 x_{1i} + \beta_2 x_{2i} + \beta_3 x_{3i} + \epsilon_i$$

and for whatever reason you want to impose the constraint that $\beta_2\beta_3 = 5$. We cannot use the `constraint` command in conjunction with `regress` because `constraint` only works with linear constraints. `nl`, however, provides an easy way out. Our constraint implies that $\beta_3 = 5/\beta_2$, so we can type

```
. nl (y = {a} + {b1}*x1 + {b2=1}*x2 + (5/{b2})*x3)
```

Here we initialized $\beta_2$ to be 1 because if the product of $\beta_2$ and $\beta_3$ is not 0, then neither of those parameters can be 0, which is the default initial value used by `nl`.