**f\_able**: Estimation of marginal effects for models with alternative variable transformations

by

Fernando Rios-Avila

Levy Economics Institute

Annandale-on-Hudson, NY USA

[friosavi@levy.org](mailto:friosavi@levy.org)

**Abstract**

margins is a powerful post-estimation command that allows the estimation of marginal effects for official and community-contributed commands, with well-defined predicted outcomes (see predict). While the use of factor variable notation allows us to easily estimate marginal effects when interactions and polynomials are used, estimation of marginal effects when other types of transformations such as splines, logs, or fractional polynomials, among others, are used remains a challenge. This paper describes how margins capabilities can be extended to analyze other variable transformations using the command f\_able.

Keywords: f\_able, margins, predict, nonlinear

1. **Introduction**

margins is a powerful post-estimation command that was introduced in Stata 11, allowing the estimation of marginal effects for all official estimation commands and any community-contributed command with a properly defined predicted outcome program (see predict). As described in Williams (2012) the introduction of margins, and its companion marginsplot (in Stata 12), provides a great tool for analyzing and providing meaningful information that is easier to interpret and describe. Furthermore, in combination with factor variable notation, also introduced in Stata 11, margins also allows users to easily estimate marginal effects when interactions and polynomials of continuous variables, and interactions with discreet variables, are used without any additional work from the user.

Despite these features, the ability of margins to estimate marginal effects is limited to the use of simple polynomials and interactions. Unless further information is provided to a command, Stata cannot identify the possible interdependencies across explanatory variables.

To the best of my knowledge, there are only two commands where this limitation is not binding. As described in Poi (2008), nlcan be used to obtain marginal effects when using functions other than interactions of polynomials, even when the model is nonlinear in parameters, but assuming an additive error. npregress series, one of the newest additions to Stata 16 nonparametric analysis, also estimates marginal effects of arbitrary transformations of the independent variables (splines, B-splines, and polynomials), based on numerical derivatives. Unfortunately, both models are based on least-squares type of estimators. Neither command can be used in the case of nonlinear models like logit or probit. In the case of npregress series, users cannot freely choose what transformations to use. The only other program that proposes a strategy that may allow obtaining marginal effects, more specifically marginal means, using transformed covariates was proposed by Royston (2013). Their command, marginscontplot uses a predefined mapping between original and transformed covariates to appropriately handle the constructed variables in margins.

This article describes a simple strategy, implemented by f\_able, that allows the estimation of marginal effects of a larger set of nonlinear transformations using margins and the option nochain [numerical noestimcheck]. Section 2 provides a review of how marginal effects should be estimated, with emphasis on some of the computational challenges that are bypassed by the use of factor notation. Section 3 provides a simple approach for the estimation of marginal effects for the case of Spline regressions and compares it to a simplified case using npregress series. Section 4 introduces three commands that prepare the data for the estimation of marginal effects for any arbitrary variable transformation and shows how f\_able can be used to estimate marginal effects using any variable transformation. Section 5 concludes.

1. **Marginal effects: Theoretical approach**

Marginal effects are useful statistics to measure the impact of a change in the independent variables will have on the dependent variable, assuming other covariates remain constant. At the beginning of most introductory econometric courses, very little emphasis is placed on understanding this concept because, for the case of linear regressions, marginal effects are typically equal to the coefficient associated with the variable analyzed. Consider the following linear regression model:

|  |  |
| --- | --- |
|  | (1) |

Under the standard assumptions of exogeneity, homoskedasticity, and correct model specifications (see Wooldridge (2016)), the coefficients of this model (equation (1)) can be estimated using Ordinary Least Squares (OLS). Under the *ceteris paribus* assumption, the marginal effect of and on could be estimated based on the thought experiment of what would the outcome be if () increases in one unit, keeping everything else constant. Mathematically, this is equivalent to obtaining the partial derivative of with respect to , which for model (1) is given by ():

However, if we lift the homoskedasticity assumption, it is more convenient to estimate marginal effects using the conditional expectation assumption. Under this assumption, we estimate the average effect that a one-unit change in or across all individuals with the same observed characteristics (conditioning on ):

This implies that for a simple linear regression (like equation(1)), where each variable appears only once, and without any transformation, marginal effects are directly identified by the estimated coefficients.

Later in the semester, students are introduced to the idea that nonlinear transformations and interactions of the independent variables can be included in the linear regression model. Because the model is still linear in parameters it can be estimated using OLS. However, additional care needs to be taken when estimating the marginal effects, as one needs to account for the interdependence of variable transformations. Consider the following model, with its conditional expectation form:

|  |  |
| --- | --- |
|  | (2a) |
|  | (2b) |

In this model, the marginal effect of on is no longer a constant, and it now depends on the values of and also of . Using calculus, the marginal effects for and in this model are:

Because marginal effects are now functions of and , one usually needs to decide what values of and use to report as the presentative marginal effect. Alternatively, one can also estimate marginal effects for all observed (or feasible) combinations of and , and summarize (or plot) those estimates as needed.[[1]](#footnote-1) The standard practice is presenting average marginal effects (AME) or the marginal effects at the mean (MEM). For the model in equation(2b) they both will be the same:

|  |  |  |
| --- | --- | --- |
| AME | MEM |  |
|  |  |  |
|  |  |  |

With this information, and assuming and are non-stochastic, standard errors associated with these marginal effects can be estimated,[[2]](#footnote-2) and the technical part of the analysis would be done.

The challenge with most software (including Stata) is that, unless additional information is provided, it may not be recognized that some variables are interrelated to each other by constructions and that the assumption of “everything else remaining constant” is incorrect. In the case of Stata, until factor variables were introduced, it could not automatically adjust for these interrelations, providing incorrect estimations of marginal effects, unless further steps were considered. Stata still fails to grasp these interrelations when we step beyond simple interactions or polynomials.

1. **Marginal effects: Empirical approach**

For this section, the dataset “Fictional data on monthly drunk driving citations” will be used and it is freely available online. Consider the following model:

|  |  |
| --- | --- |
|  | (3) |

Before Stata 11 and factor notation, to estimate a model like this, all variables would need to be created before including them in the model. For example, creating a variable named fines2 to be equal to :

webuse dui,clear

gen fines2=fines^2

and fit the model using the command regress:

regress citations fines fines2

Source | SS df MS Number of obs = 500

-------------+---------------------------------- F(2, 497) = 189.57

Model | 20750.38 2 10375.19 Prob > F = 0.0000

Residual | 27200.458 497 54.7292917 R-squared = 0.4327

-------------+---------------------------------- Adj R-squared = 0.4305

Total | 47950.838 499 96.0938637 Root MSE = 7.3979

------------------------------------------------------------------------------

citations | Coef. Std. Err. t P>|t| [95% Conf. Interval]

-------------+----------------------------------------------------------------

fines | -47.10883 7.691611 -6.12 0.000 -62.22091 -31.99674

fines2 | 1.98084 .3885844 5.10 0.000 1.21737 2.744311

\_cons | 293.0067 37.94286 7.72 0.000 218.4585 367.5549

-------------------------------------------------------------------------------

Using margins to calculate marginal effect of fines on citations would produce the wrong answer since fines2 would not be handled correctly because there is no way for Stata to know that fines2=fines^2.

margins, dydx(fines)

Average marginal effects Number of obs = 500

Model VCE : OLS

Expression : Linear prediction, predict()

dy/dx w.r.t. : fines

------------------------------------------------------------------------------

| Delta-method

| dy/dx Std. Err. t P>|t| [95% Conf. Interval]

-------------+----------------------------------------------------------------

fines | -47.10883 7.691611 -6.12 0.000 -62.22091 -31.99674

------------------------------------------------------------------------------

However, average marginal effects can be derived by hand by taking the derivative of equation 3 with respect to fines, using the average value of fines as the point of reference, and using lincom to calculate the marginal effects and standard errors:

sum fines, meanonly

lincom \_b[fines]+2\*\_b[fines2]\*`r(mean)'

( 1) fines + 19.7904\*fines2 = 0

------------------------------------------------------------------------------

citations | Coef. Std. Err. t P>|t| [95% Conf. Interval]

-------------+----------------------------------------------------------------

(1) | -7.907201 .4236816 -18.66 0.000 -8.739629 -7.074773

------------------------------------------------------------------------------

Since the introduction of Stata 11, estimating marginal effects for a model like this is much simpler. Using factor notation, we add the squared parameter, and let margins handle the interaction on its own:

qui:regress citations fines c.fines#c.fines

margins, dydx(fines)

Average marginal effects Number of obs = 500

Model VCE : OLS

Expression : Linear prediction, predict()

dy/dx w.r.t. : fines

------------------------------------------------------------------------------

| Delta-method

| dy/dx Std. Err. t P>|t| [95% Conf. Interval]

-------------+----------------------------------------------------------------

fines | -7.907201 .4236816 -18.66 0.000 -8.739629 -7.074773

------------------------------------------------------------------------------

I suspect that the methodology behind the code for margins and factor notation “taught” Stata how to take derivatives of polynomials. In other words, Stata recognizes that when there is an expression like “c.var1#c.var1”, internally it “knows” the analytical derivative is 2\*c.var1. Thus, margins simply uses this information to handle the squared parameter (c.fines#c.fines), before providing you with the result.

While this has been a big improvement for a better understanding of marginal effects, it does have its limitations. For example, margins would not be able to estimate marginal effects for the following models:

|  |  |
| --- | --- |
|  | (4) |

Even though, mathematically, the average marginal effects (AME), and marginal effects at the means (MEM), can be derived directly using the sample means:

|  |  |  |
| --- | --- | --- |
|  | AME | MEM |
| Model 1: |  |  |
| Model 2: |  |  |
| Model 3: |  |  |

Which can be used to estimate the average marginal effects by hand. Now, concentrate on the estimation of the average marginal effects:

. webuse dui, clear

(Fictional data on monthly drunk driving citations)

. gen i\_fines=1/fines

. gen ni\_fines2=-1/fines^2

. gen fines05=fines^.5

. gen i\_fines05=0.5\*fines^-.5

. gen fines\_99=max((fines-9.9),0)

. gen dfines\_99=fines>9.9

\* model 1

. qui:regress citations i\_fines

. sum ni\_fines2, meanonly

. lincom \_b[i\_fines]\*`r(mean)'

( 1) - .0104108\*i\_fines = 0

------------------------------------------------------------------------------

citations | Coef. Std. Err. t P>|t| [95% Conf. Interval]

-------------+----------------------------------------------------------------

(1) | -8.091099 .4224389 -19.15 0.000 -8.921081 -7.261117

------------------------------------------------------------------------------

\* model 2

. qui:regress citations fines05

. sum i\_fines05, meanonly

. lincom \_b[fines05]\*`r(mean)'

( 1) .1593264\*fines05 = 0

------------------------------------------------------------------------------

citations | Coef. Std. Err. t P>|t| [95% Conf. Interval]

-------------+----------------------------------------------------------------

(1) | -8.010351 .4314167 -18.57 0.000 -8.857972 -7.162729

------------------------------------------------------------------------------

\* model 3

. qui:regress citations fines fines\_99

. sum dfines\_99, meanonly

. lincom \_b[fines]+\_b[fines\_99]\*`r(mean)'

( 1) fines + .5\*fines\_99 = 0

------------------------------------------------------------------------------

citations | Coef. Std. Err. t P>|t| [95% Conf. Interval]

-------------+----------------------------------------------------------------

(1) | -7.926694 .4271729 -18.56 0.000 -8.765981 -7.087407

------------------------------------------------------------------------------

For these models one also has the option of using nl, to estimate the marginal effects, which requires less work:[[3]](#footnote-3)

qui:nl (citations = {b0}+{b1}/fines), variable(fines)

margins, dydx(fines)

qui:nl (citations = {b0}+{b1}\*fines^.5), variable(fines)

margins, dydx(fines)

qui:nl (citations = {b0}+{b1}\*fines+{b2}\*max((fines-9.9),0)), variable(fines)

margins, dydx(fines)

This flexibility of nl to estimate marginal effects with non-standard transformations motivates the question: why can nl “correctly” estimate marginal effects, whereas regress can’t? The answer is rather simple, we are not using a constructed variable in the model, instead, we are using the original variable and allowing nl to construct the new variable.

This means that, while we see this model is being estimated:

citations = {b0}+{b1}\*fines^.5

What may be happening in the background is different. Stata identifies which elements of this code are parameters to be estimated (those within brackets), and which elements need to be created (fines^.5), before fitting the model. In other words, Stata is estimating the following:

citations = {b0}+{b1}\*\_\_000000

where \_\_000000 is a temporary variable, you never see, that was constructed as fines^.5. The difference is that nl knows that \_\_000000= fines^.5.

How is it that nl knows what the derivative of is ? The answer is nl does not know. The only type of analytical derivatives Stata may know is when interactions are used, specifically factor notations. However, because Stata “remembers” how a variable was constructed, it can use numerical derivatives to make a reasonable approximation for the analytical derivative.

For example, the numerical derivative for the transformation can be approximated as follows:

This expression is surprisingly accurate.[[4]](#footnote-4) For this example, when the largest absolute difference between the numerical and analytical derivative is 0.000423, whereas when , the largest difference is 6.58e-09. This implies that margins does not need to know how to obtain analytical derivatives, since it can use numerical derivatives instead, and use this information to estimate the appropriate marginal effects.

1. **f\_able: marginal effects for arbitrary variable transformations**

The previous sections described how marginal effects should be estimated and how to use that information to obtain AME or MEM by comparing the step-by-step procedure with what margins does. It also details how nl is able to estimate marginal effects for variable transformations other than interactions using numerical derivatives to approximate the analytical derivatives. This implies that Stata already has the capability to estimate marginal effects when transformations other than interactions and polynomials are used in a model. However, to be able to expand those capabilities to any linear and non-linear model, we need to how to “tell” Stata that some variables are based on constructions from others, so Stata can use this information to correctly estimate marginal effects.

In this section I first sketch the basic structure of the programs required to solve this problem, and second described the commands within f\_able, that address this problems.

* 1. **f\_able: Basics**

One solution to pass information about variable dependencies to Stata is to use the variable label to “store” the transformation used to create the variable of interest. To automate this procedure, two small programs are proposed that “wrap” around Stata’s built-in commands generate and replace:

program fgen

syntax newvarname =/exp [if] [in]

gen `typelist' `varlist'=`exp'

label var `varlist' "`exp'"

end

program frep

syntax varname =/exp [if] [in]

replace `varlist'=`exp'

label var `varlist' "`exp'"

end

These two commands have one purpose, when a new variable is created, it will label it with the expression used to create it, and if the values are replaced, it will change the label to the new expression used.

Consider model 3 from equation (4). The variable of interest can be created using the command fgen, but I created it with an intentional mistake:

. fgen fines2=max(fines-9,0)

. describe fines2

storage display value

variable name type format label variable label

------------------------------------------------------------------

fines2 float %9.0g max(fines-9,0)

However, I can rectify it using the command frep and replace the values fines2 with the correct content:

. frep fines2=max(fines-9.9,0)

(420 real changes made)

. describe fines2

storage display value

variable name type format label variable label

-----------------------------------------------------------------

fines2 float %9.0g max(fines-9.9,0)

In addition to setting up the data for the next step of the estimation of marginal effects, these commands may be useful for tracking how variables are created or modified.

The second step is to explicitly tell Stata that a particular variable is constructed based on other variables in the model, and that when margins is called, the constructed variable needs to be updated every time the original variable changes. This can be done with the two following programs:

program f\_able, eclass

syntax, [\* NLvar(varlist)]

\_ms\_dydx\_parse `nlvar'

if "`e(predict\_old)'"=="" {

ereturn local predict\_old `e(predict)'

ereturn local predict f\_able\_p

}

foreach i of varlist `nlvar' {

local fnc:variable label `i'

ereturn hidden local \_`i' `fnc'

}

ereturn local nldepvar `nlvar'

end

program f\_able\_p

syntax newvarname [if] [in], [\*]

local idepvar `e(nldepvar)'

foreach i of local idepvar {

tempvar \_`i'

qui:clonevar `\_`i''=`i'

qui:recast double `i'

qui:replace `i'=`e(\_`i')'

}

`e(predict\_old)' `0'

foreach i of local idepvar {

if "`i'"!="\_cons" {

qui:replace `i'=`\_`i''

}

}

end

The first program, f\_able, does three things. First, it adds information to any previously estimated model indicating which variables are constructed variables using e(nldepvar). Second, it adds hidden macros with information regarding the data transformation used.[[5]](#footnote-5) Lastly, it redirects predict from the original e(predict) command to the one defined below as f\_able\_p, but keeping the original information in e(predict\_old).

The second program, f\_able\_p, has only one purpose, updating the constructed variables, identified in e(nldepvar), before proceeding to obtain the predicted values appropriate for the estimated command e(predict\_old), using the information previously stored in the hidden macros.

With these two pieces of code, the last step is to call margins for the estimation of the marginal effects, using the option nochainrule.[[6]](#footnote-6) This is an option often overlooked when using official Stata commands. However, as its help file indicates:

“nochainrule is safer because it makes no assumptions about how the parameters and covariates join to form the response.”

This implies that when a model like the following is estimated:

regress citations fines fines2

f\_able, nl(fines2)

Marginal effects will be calculated using the coefficient of fines, and the coefficient of fines2 times the numerical derivative of fines2 with respect to fines, if fines2 was declared as a constructed variable. Let see how this works.

. qui:regress citations fines fines2

. f\_able, nlvar(fines2)

. margins, dydx(fines) nochainrule

Average marginal effects Number of obs = 500

Model VCE : OLS

Expression : Fitted values, predict()

dy/dx w.r.t. : fines

------------------------------------------------------------------------------

| Delta-method

| dy/dx Std. Err. z P>|z| [95% Conf. Interval]

-------------+----------------------------------------------------------------

fines | -7.926694 .4271729 -18.56 0.000 -8.763937 -7.089451

------------------------------------------------------------------------------

The first line estimates the model with citations as a dependent variable and fines and fines2 as independent variables. fines2 is the one we defined previously as max(fines-9.9,0). The second line calls f\_able to identify that fines2 is a constructed variable, and the last step estimates the marginal effects using nochainrule. The results are the same as those done by hand, or those using nl command. They can also be replicated using the npregress series.

. npregress series citations fines, spline(1) knots(1)

Computing approximating function

Computing average derivatives

Linear-spline estimation Number of obs = 500

Number of knots = 1

------------------------------------------------------------------------------

| Robust

citations | Effect Std. Err. z P>|z| [95% Conf. Interval]

-------------+----------------------------------------------------------------

fines | -7.926694 .4772213 -16.61 0.000 -8.86203 -6.991357

------------------------------------------------------------------------------

Note: Effect estimates are averages of derivatives.

Notice that while the point estimates are the same, the standard errors produced by npregress series are somewhat larger than those produced with nl or with the proposed strategy.

We can also compare f\_able to the output we could obtain using factor notation:

qui:regress citations c.fines##c.fines##c.fines

. margins, dydx(fines)

Average marginal effects Number of obs = 500

Model VCE : OLS

Expression : Linear prediction, predict()

dy/dx w.r.t. : fines

------------------------------------------------------------------------------

| Delta-method

| dy/dx Std. Err. t P>|t| [95% Conf. Interval]

-------------+----------------------------------------------------------------

fines | -7.928817 .4226225 -18.76 0.000 -8.759168 -7.098465

------------------------------------------------------------------------------

. frep fines2=fines^2

(500 real changes made)

. fgen fines3=fines^3

. qui:regress citations fines fines2 fines3

. f\_able, nlvar(fines2 fines3)

. margins, dydx(fines) nochainrule

Average marginal effects Number of obs = 500

Model VCE : OLS

Expression : Fitted values, predict()

dy/dx w.r.t. : fines

------------------------------------------------------------------------------

| Delta-method

| dy/dx Std. Err. z P>|z| [95% Conf. Interval]

-------------+----------------------------------------------------------------

fines | -7.928817 .4226223 -18.76 0.000 -8.757141 -7.100492

------------------------------------------------------------------------------

This method replicates the results when using factor notation up to five decimal places, but some degree of precision is lost due to forced use of numerical derivatives.

Lastly, a more challenging estimation that combines the use of factor notation with f\_able and compares it to the output of the npregress series.

. npregress series citations fines i.csize, spline(2) knots(1)

Computing approximating function

Computing average derivatives

Quadratic-spline estimation Number of obs = 500

Number of knots = 1

------------------------------------------------------------------------------------

| Robust

citations | Effect Std. Err. z P>|z| [95% Conf. Interval]

-------------------+----------------------------------------------------------------

fines | -7.590817 .3533786 -21.48 0.000 -8.283427 -6.898208

|

csize |

(medium vs small) | 5.48074 .5827045 9.41 0.000 4.33866 6.622819

(large vs small) | 10.69879 .6043375 17.70 0.000 9.514311 11.88327

------------------------------------------------------------------------------------

Note: Effect estimates are averages of derivatives for continuous covariates and averages of contrasts for factor covariates.

The same can be replicated using regress and f\_able:

. webuse dui,clear

. fgen double fines2=fines^2

. fgen double fines3=max(fines-9.9,0)^2

. qui:regress citations c.(fines fines2 fines3)##i.csize

. f\_able, nlvar(fines2 fines3)

. margins, dydx(fines csize) nochainrule

Average marginal effects Number of obs = 500

Model VCE : OLS

Expression : Fitted values, predict()

dy/dx w.r.t. : fines 2.csize 3.csize

------------------------------------------------------------------------------

| Delta-method

| dy/dx Std. Err. z P>|z| [95% Conf. Interval]

-------------+----------------------------------------------------------------

csize |

medium | 5.48074 .6522747 8.40 0.000 4.202305 6.759174

large | 10.69879 .617192 17.33 0.000 9.489116 11.90846

|

fines | -7.590817 .3350831 -22.65 0.000 -8.247568 -6.934067

------------------------------------------------------------------------------

Note: dy/dx for factor levels is the discrete change from the base level.

Once again, this replicates the output with very small differences in the point estimates, but with some differences in terms of the standard errors. This may be a result of degrees of freedom adjustment used in npregress.

We can also use this strategy for models other than OLS. Consider the dataset mksp2. Similar to the example provided in the help file for the command mkspline, four variables are created to allow for linear splines with 4 knots. A logit model of the outcome against dosage and the constructed variables is estimated. margins needs to include the options nochain and numerical. The predicted probabilities and the marginal effects across various values of dosage are produced:

webuse mksp2, clear

fgen dos1=max(dosage-17.5,0)

fgen dos2=max(dosage-36.5,0)

fgen dos3=max(dosage-55.5,0)

fgen dos4=max(dosage-81.5,0)

qui:logit outcome dosage dos1 dos2 dos3 dos4

f\_able, nl(dos1 dos2 dos3 dos4)

qui:margins , nochain numerical at(dosage=(0(2)100))

marginsplot, name(m1)

qui:margins , dydx(dosage) nochain numerical at(dosage=(0(2)100))

marginsplot, name(m2)

graph combine m1 m2, xsize(8) scale(1.5)

**Figure 1. Predicted probabilities and Marginal effects**

****

* 1. f\_able package

In the previous subsection, I sketched the basic code that can be used to stored information in regression outputs so that it can be used by margins to estimate marginal effects after most linear and nonlinear models. In this subsection, I describe all the programs that are part of f\_able and can be used to address some limitations of the strategy.

fgen newvar =exp

frep oldvar =exp

fgen and frep can be used to create a new variable, or replace an existing variable using any expression that is valid with generate. fgen creates a variable in double format to reduce problems of precision with the constructed variables. If the expression is longer than 75 characters, it assigns the label "See note", and a variable note will store the used expression.

f\_able , nlvar(varlist)

f\_able is used to declare which variables in the model specification should be considered as “constructed” variables. All variables in nlvar, which should be present in the previous model specification, are stored in a macro named e(nldepvar). After using f\_able, marginal effects with respect to the constructed variables are no longer possible unless f\_able\_reset is used. A list of hidden macros named e(\_varname) are also stored containing the information of how each variable (varname) was created.

f\_able\_reset, restores all the post the estimation results to the state previous to using f\_able.

f\_symev and f\_symrv are commands that force the variance covariance matrices stored in e(V) and r(V), respectively, to be symmetric when *nonsymmetric* warning appears. The warning appears as a consequence of the precision loss and forced numerical derivatives, which creates variance covariance matrixes that are nonsymmetric. This often occurs after using margins for multiple points of reference.

1. **Conclusion**

This article describes how marginal effects can be estimated using analytical derivatives, as well as numerical derivatives. It introduced two small programs, within the f\_able package, that enable margins to estimate marginal effects when using transformations beyond variable interactions and polynomials.

These commands can be used to estimate marginal effects for many official Stata commands as well as other community-contributed commands that can produce sensible predicted outcomes. The strategy does have three limitations: first, the estimated marginal effects depend on the precision of the forced numerical derivatives; second, it requires the original variable to be present in the model specification so that marginal effects can be used; and third, because f\_able is forcing margins to do something it is not meant to do, one may experience difficulties estimating marginal effects and standard errors when the request is particularly complex.

While the limitation regarding the precision of the estimates is unavoidable, the other limitations can be circumvented to some extent. First, the original variable can be added to the list of explanatory variables using “o.”. This omits the original variable from the estimation but keeps it in the list of explanatory variables, allowing the estimation of margins.

The last limitation can be addressed with careful troubleshooting. While the option nochain forces margins to use numerical derivatives for the estimation of marginal effects after most commands, some commands like probit, logit, and poisson require using the options nochain numerical. If point estimates appear as missing, the option noestimcheck can be used to bypass some of the safety checks in margins that produce this error. Finally, if standard errors are missing when estimating marginal effects for multiple points of interest, it may be because the estimated variance-covariance matrix created after margins is nearly symmetric. For those cases f\_symev and f\_symrv can be used to force symmetry on the variance-covariance matrix produced by margins, so one can report standard errors or plot confidence intervals after marginsplot.

1. **References**

Poi, B. P. 2008. "Stata tip 58: nl is not just for nonlinear models." *Stata Journal* 8 (1):139-141.

Royston, Patrick. 2013. "marginscontplot: Plotting the marginal effects of continuous predictors." *The Stata Journal* 13 (3):510-527.

Williams, Richard. 2012. "Using the margins command to estimate and interpret adjusted predictions and marginal effects." *The Stata Journal* 12 (2):308-331.

Wooldridge, Jeffrey M. 2016. *Introductory econometrics: A modern approach*: Nelson Education.

**About the author**

Fernando Rios-Avila is a research scholar at Levy Economics Institute of Bard College under the Distribution of Income and Wealth program. His research interests include applied econometrics, labor economics, and poverty and inequality.

1. It should be noted that the command margins can generate marginal effects for all available data using the undocumented option gen(). I want to thank an anonymous referee for pointing this out. [↑](#footnote-ref-1)
2. A detailed explanation of how standard errors are estimated using margins can be found <https://www.stata.com/support/faqs/statistics/compute-standard-errors-with-margins/> [↑](#footnote-ref-2)
3. For this to work, we need to indicate to nl that fines is a variable in the model. Outputs are identical to the ones produced by hand. [↑](#footnote-ref-3)
4. As a matter of fact, as indicated on its help file, the function deriv() in mata uses this approximation to compute numerical derivatives. [↑](#footnote-ref-4)
5. While this step is unnecessary in most cases I have considered, in some examples margins “drops” the variable label information. In those cases, storing the information as part of the estimation commands seems to be better alternative. [↑](#footnote-ref-5)
6. Some commands, like logit, probit and poisson, also require to include the option numerical , to use numerical derivatives. [↑](#footnote-ref-6)