f_able: Estimation of marginal effects with transformed covariates

Taking Margins a step further

Rios-Avila, Fernando[1]

[1]friosavi@levy.org
Levy Economics Institute

Stata Conference, July 2020

Table of Contents

Table of Contents

Introduction

- Marginal effects tells us how a dependent variable (outcome) y changes when an independent variable x changes, assuming everything else constant (e and z's).

$$y = b_0 + b_1 x + b_2 z + e$$

- For linear models, with no interactions or polynomials, marginal effects are equal to their coefficients:

$$\frac{dy}{dx} = b_1 \& \frac{dy}{dz} = b_2$$

- However, when there are interactions, polynomials, or other transformations, further work is needed.

Table of Contents

Estimating Marginal effects

- When interactions or polynomials are used, marginal effects should be obtained estimating equation derivatives:

$$y = b_0 + b_1 x + b_2 x^2 + b_3 z + b_4 zx + e$$

$$\frac{dy}{dx} = b_1 + 2b_2 x + b_4 z$$

$$\frac{dy}{dz} = b_3 + b_4 x$$

- Main difference with simple linear model?
  - Marginal effects no longer constant
  - Coefficients alone are not useful
  - Derivatives are needed to obtain the effects.

Estimating Marginal effects

How to proceed in this case? what to report? There are many options:

$$AvgME = E(\frac{dy}{dx})$$

$$MEatMean = \frac{dy}{dx}|X = \bar{x}; z = \bar{z}$$

$$MEatvalues = \frac{dy}{dx}|X = X; z = Z$$

Or report "ALL" effects for each observation in the data. Then "simply" estimate SE.

Table of Contents

Empirical Estimation of Marginal effects

- Before Stata 11, estimation of marginal effects for models with interactions was "hard".
- You needed to create the variables "by hand", and adjust marginal effects on your own:

```
. webuse dui, clear
. gen fines2=fines*fines
. reg citations fines fines2
. sum fines2
. lincom _b[fines]+2*_b[fines2]*`r(mean)'
```

- Otherwise, using the old -mfx- or the new -margins- would give you incorrect results.
- why? because Stata does not recognize that $fines2 = fines^2$.(much less how to obtain the derivative)
- The solution, Teach Stata how to do it.

Margins and Factor notation, and limitations

- Stata 11 introduced the use of factor notation, and margins.
- Factor notation (c. # i.) facilitates adding interactions to models, so that correct marginal effects can be estimated using `margins`
- Marginal effects for the previous model can be easily estimated:

```
. webuse dui, clear
. reg citations fines c.fines#c.fines
(where c.fines#c.fines=fines^2)
. margins, dydx(fines)
```

- Internally, `margins` understand c.fines#c.fines depends on fines. (And probably estimates analytical derivatives to obtain the ME).
- but what if you want to use other transformations?: $fines^{.5}$, $log(fines)$, *splines*, *fracpoly*, etc
- *Impossible*, or is it?

Table of Contents

The Limitations of `margins`

- For the previous examples, `margins` after regress does not work.
- However there are other alternatives:
- `npregress` estimates full nonparemetric regressions using kernel or series methods:

```
. npregress kernel citations fines
. npregress series citations fines
```

- `nl` can also be used for this (Poi 2008)

```
. nl (citations={a0}+{a1}*fines^0.5), variable(fines)
. margins, dydx(fines)
```

- And there is one community-contributed commands that can be used for plotting this type of effects (`marginscontplot` by Royston (2013)).

Table of Contents

Beyond factor notation

- The way `nl`, and npregress works shows that Stata can estimate marginal effects with variable transformations other than interactions... It just doesn't know it yet
- Three problems need to be address for Stata to do this:
  - Store information of how a variable is created.
  - Identify that a variable is a *constructed* variable.
  - Use that information to obtain partial effects.
- Here is where `f_able` helps solving these problems.

f_able package: fgen and frep

- To solve the first problem, I propose fgen and frep. These commands are wrappers around generate and replace that stores how the variable was generated, as a label or note.

```
. ssc install f_able
. qui:fgen fines2=fines^2
. describe fines2
              storage   display    value
variable name   type    format     label      variable label
------------------------------------------------------------------
fines2          double  %10.0g                fines^2

. qui:frep fines2=fines*fines
. describe fines2
              storage   display    value
variable name   type    format     label      variable label
------------------------------------------------------------------
fines2          double  %10.0g                fines*fines
```

f_able package: f_able

- To solve the second problem, I propose f_able. This is a post estimation command that identifies what variables in a model are "constructed" variables, adding information to any previously estimated model, and redirecting the predict sub-command to f_able_p.

```
. qui:reg citations fines fines2
. f_able, nl(fines2)
. ereturn list, all
scalars: (omitted)
macros: (other macros omitted)
            e(nldepvar) : "fines2"
             e(predict) : "f_able_p"
         e(predict_old) : "regres_p"
Hidden macros: (other hidden macros omitted)
             e(_fines2) : "fines*fines"
```

f_able package: f_able_p

- To solve the third problem, I propose `f_able_p`. This passive command uses the information left by `f_able` to update all constructed values when the original variable changes, before using predict for the margins estimation.
- Only difference, when calling `margins` we need to include the option `nochain`, so numerical derivatives are used.

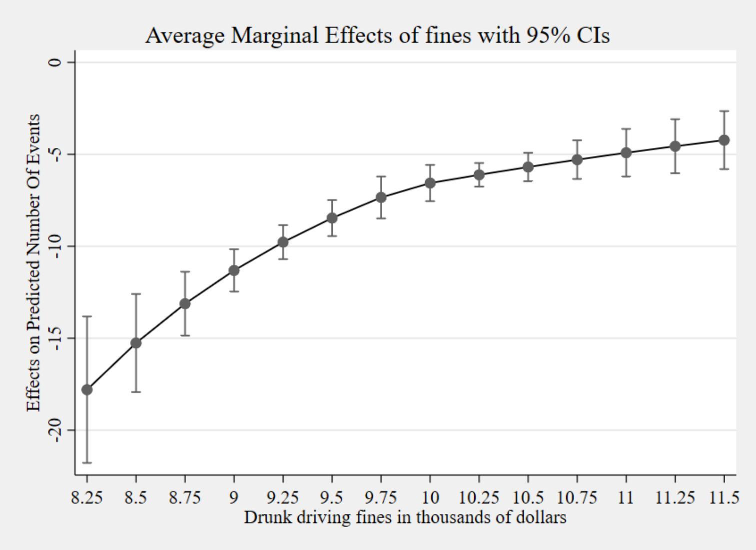
```
. qui:reg citations fines fines2
. f_able, nl(fines2)
. margins, dydx(fines) nochain
Average marginal effects                        Number of obs     =        500
Model VCE     : OLS
Expression    : Fitted values, predict()
dy/dx w.r.t. : fines
------------------------------------------------------------------------------
             |            Delta-method
             |      dy/dx   Std. Err.      z    P>|z|     [95% Conf. Interval]
-------------+----------------------------------------------------------------
       fines |  -7.907201   .4236816   -18.66   0.000    -8.737602    -7.0768
------------------------------------------------------------------------------
```

Example: poisson with quadratic Spline

A Small example using a nonlinear model (poisson) with a quadratic spline with 1 knot. Main difference, after poisson, margins need options "nochain and numerical".

```
webuse dui, clear
fgen fines2=fines^2
fgen fines3=max(fines-9.9,0)^2
qui:poisson citations fines fines2 fines3
f_able, nl(fines2 fines3)
* Marginal effects
margins, dydx(fines) at(fines=(8.25 (.25) 11.5)) ///
nochain numerical plot
* Predicted means
margins, at(fines=(8.25 (.25) 11.5)) ///
nochain numerical plot
```

Avg Marginal effects



Predictive Margins

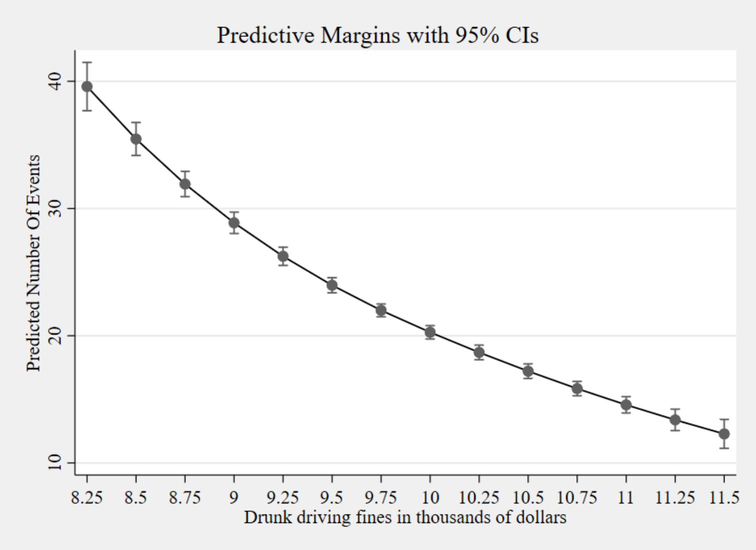


Table of Contents

Conclusions

- This presentation introduces the package `f_able`, as a post estimation command that enables `margins` to estimate marginal effects with transformed covariates
- While the strategy has some limitations, it can provide researchers with a simple tool to make the best of more flexible model specifications.

For more examples see the help file "ssc install f_able"
Working paper available at: https://bit.ly/rios_fable

Thank you!

References

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

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

Rios-Avila, Fernando. (forthcoming). "f_able: Estimation of marginal effects for models with alternative variable transformations". The Stata Journal