

From the help desk: hurdle models

Allen McDowell
Stata Corporation

Abstract. This article demonstrates that, although there is no command in Stata for fitting hurdle models, the parameters of a hurdle model can be estimated in Stata rather easily using a combination of existing commands. We also include a likelihood evaluator to be used with Stata's `ml` facilities to illustrate how to fit a hurdle model using `ml`'s `cluster()`, `svy`, and `constraints()` options.

Keywords: st0040, hurdle model

1 Introduction to hurdle models

A hurdle model is “a modified count model in which the two processes generating the zeros and the positives are not constrained to be the same” (Cameron and Trivedi 1998). Mullahy (1986) states, “The idea underlying the hurdle formulations is that a binomial probability model governs the binary outcome of whether a count variate has a zero or a positive realization. If the realization is positive, the “hurdle is crossed”, and the conditional distribution of the positives is governed by a truncated-at-zero count data model.” Following Mullahy, but with a change in notation, let $F_1(\beta_1)$ represent the probability that the hurdle is crossed, and let $f_2(y, \beta_2)/F_2(\beta_2)$, $y \in \Gamma_+$ be the conditional distribution of the positives, where f_2 satisfies $\sum_{y \in \Gamma_+} f_2(y, \beta_2) = 1$, F_2 is the summation of f_2 on the support of the conditional density (i.e., the truncation normalization), and $y \in \Gamma_+ = \{1, 2, 3, \dots\}$. The general form of the hurdle model likelihood function is then

$$L = \prod_{i \in \Omega_0} \{1 - F_1(\beta_1)\} \prod_{i \in \Omega_1} \frac{\{f_2(y_i, \beta_2) F_1(\beta_1)\}}{F_2(\beta_2)}$$

where $\Omega_0 = \{i | y_i = 0\}$, $\Omega_1 = \{i | y_i \neq 0\}$, and $\Omega_0 \cup \Omega_1 = \{1, 2, \dots, N\}$. Taking the natural logarithm of both sides and rearranging terms, we see that the log likelihood can be written as

$$\ln(L) = \sum_{i \in \Omega_0} \ln\{1 - F_1(\beta_1)\} + \sum_{i \in \Omega_1} \ln\{F_1(\beta_1)\} + \sum_{i \in \Omega_1} \left[\ln\{f_2(y_i, \beta_2)\} - \ln\{F_2(\beta_2)\} \right]$$

Since the likelihood function is separable with respect to the parameter vectors β_1 and β_2 , the log likelihood can always be written as the sum of the log likelihoods from two separate models: a binomial probability model and a truncated-at-zero count model. As such, the hurdle model log likelihood can always be maximized, without loss of information, by maximizing the two components separately. This feature of hurdle models allows us to fit hurdle models in two separate steps using existing Stata commands. For example, we could use `cloglog`, `logit`, `probit`, or `glm` to model the

binomial probability model, and `trpois0` (Hilbe 1999) or `trbin0` (Hilbe 1999) to model the truncated count model. The procedure is demonstrated below for a hurdle model consisting of a complementary log-log binomial probability model and a truncated-at-zero Poisson count model. A likelihood evaluator for the same model is also included to illustrate how to fit the same hurdle model using `ml`'s facilities so that we can compare the statistical results from the two procedures and extend our modeling capabilities for hurdle models by including `ml`'s `cluster()`, `svy`, and `constraints()` options.

2 The Poisson hurdle model specification

We start with the binomial process, which determines whether the dependent variable takes on the value zero or a positive value. The probability mass function is

$$\Pr(Y = y) = \begin{cases} \pi, & y = 0 \\ 1 - \pi, & y = 1, 2, 3, \dots \end{cases}$$

The zero-truncated Poisson process has probability mass function

$$\Pr(Y = y | Y \neq 0) = \begin{cases} \frac{\lambda^y}{(e^\lambda - 1)y!}, & y = 1, 2, 3, \dots \\ 0, & \text{otherwise} \end{cases}$$

Thus, the unconditional probability mass function for Y is

$$\Pr(Y = y) = \begin{cases} \pi, & y = 0 \\ (1 - \pi) \frac{\lambda^y}{(e^\lambda - 1)y!}, & y = 1, 2, 3, \dots \end{cases}$$

and the log likelihood for the t^{th} observation, assuming the observations are independently and identically distributed, is

$$\ln L(\pi_i, \lambda_i, y_i) = \begin{cases} \ln \pi_i, & y = 0 \\ \ln \left\{ (1 - \pi_i) \frac{\lambda_i^{y_i}}{(e^{\lambda_i} - 1)y_i!} \right\}, & y = 1, 2, 3, \dots \end{cases}$$

If we model π_i using the complementary log-log link and λ_i using the log link, with a little algebra we have

$$\pi_i = e^{-e^{\mathbf{x}_i \boldsymbol{\beta}_1}}$$

and

$$\lambda_i = e^{\mathbf{x}_i \boldsymbol{\beta}_2}$$

Thus, the log likelihood can be written

$$\begin{aligned}
 \ln L &= \ln \left\{ \prod_{i \in \Omega_0} \left(e^{-e^{\mathbf{x}_i \beta_1}} \right) \prod_{i \in \Omega_1} \left(1 - e^{-e^{\mathbf{x}_i \beta_1}} \right) \prod_{i \in \Omega_1} \frac{e^{y_i \mathbf{x}_i \beta_2}}{(e^{e^{\mathbf{x}_i \beta_2}} - 1) y_i!} \right\} \\
 &= \left\{ \sum_{i \in \Omega_0} -e^{\mathbf{x}_i \beta_1} + \sum_{i \in \Omega_1} \ln(1 - e^{-e^{\mathbf{x}_i \beta_1}}) \right\} \\
 &\quad + \left\{ \sum_{i \in \Omega_1} y_i \mathbf{x}_i \beta_2 - \sum_{i \in \Omega_1} \ln(e^{e^{\mathbf{x}_i \beta_2}} - 1) - \sum_{i \in \Omega_1} \ln(y_i!) \right\} \\
 &= \ln\{L_1(\beta_1)\} + \ln\{L_2(\beta_2)\}
 \end{aligned}$$

We can see that the log likelihood describes the sum of a log likelihood for the binary outcome model, $\ln L_1(\beta_1)$, and a log likelihood for a truncated-at-zero Poisson model, $\ln L_2(\beta_2)$. As indicated above, the β_1 and β_2 vectors of parameters are separable. This separability implies that the Hessian will be block diagonal so that the covariances between β_1 and β_2 are zero. Therefore, we will not lose information if we fit a hurdle model by estimating the parameters of the binomial probability model separately from the parameters of the truncated Poisson model.

3 Fitting hurdle models in two steps

To demonstrate that we can use existing commands to fit a hurdle model, let's begin by simulating some data that follow a hurdle process as described above. First, we generate two normally distributed random variables that will serve as the independent variables in the model.

```

. clear
. set obs 2000
. set seed 1000
. generate x1=invnorm(uniform())
. generate x2=invnorm(uniform())

```

Using those newly generated covariates, we next generate a variable that follows a truncated-at-zero process. To do this, we make use of the user-written command `rndpoix` (Hilbe and Linde-Zwirble 1998), which generates a variable from a Poisson process and names it `xp`. Once the variable `xp` has been generated, the observations containing zeros can be dropped, leaving us with a truncated-at-zero Poisson random variable.

```

. generate lambda=exp(.2*x1 + .7*x2 + 1)
. rndpoix lambda

```

```
. drop if xp == 0
```

Next, we generate a variable using the binomial model described above.

```
. generate pi = exp(-exp(.9*x1 + .1*x2 + .2))
. generate bernoulli = uniform()>pi
```

Finally, we repopulate the truncated-at-zero poisson variable with zeros if the “hurdle” has not been crossed.

```
. replace xp = 0 if bernoulli==0
. rename xp y
. keep in 1/1000
```

Now that we have generated the data, we can proceed to fit a hurdle model in two steps.

Step 1: First, we can estimate the parameters of the binomial probability model using Stata’s `cloglog` command.

```
. cloglog bernoulli x1 x2, nolog
Complementary log-log regression
```

Number of obs	=	1000
Zero outcomes	=	315
Nonzero outcomes	=	685
LR chi2(2)	=	319.71
Prob > chi2	=	0.0000

Log likelihood = -463.18843

bernoulli	Coef.	Std. Err.	z	P> z	[95% Conf. Interval]	
x1	.9287196	.0647568	14.34	0.000	.8017986	1.055641
x2	.0998809	.0497777	2.01	0.045	.0023184	.1974435
_cons	.1495369	.0491899	3.04	0.002	.0531264	.2459474

Step 2: Now, we can estimate the parameters of the truncated Poisson model with the user-written command `trpois0` (Hilbe 1999).

```
. trpois0 y x1 x2 if y > 0, nolog
0-Truncated Poisson Estimates
```

Number of obs	=	685
Model chi2(2)	=	498.18
Prob > chi2	=	0.0000
Pseudo R2	=	0.2171

Log Likelihood = -898.3545976

y	Coef.	Std. Err.	z	P> z	[95% Conf. Interval]	
x1	.2472253	.0363057	6.81	0.000	.1760675	.3183831
x2	.6801573	.0319771	21.27	0.000	.6174832	.7428313
_cons	.0166542	.0528191	0.32	0.753	-.0868694	.1201777

Notice that the 95% confidence intervals contain the population parameters we used to generate the data.

4 Fitting hurdle models in one step with ml

Since there is no loss of information when fitting a hurdle model in two steps, if we estimate the parameters jointly using `ml`, we should obtain identical estimates of the model parameters and their variances. Furthermore, the log likelihood obtained from estimating with `ml` should equal the sum of the log likelihoods obtained from fitting the binomial probability model and the truncated Poisson models separately. In addition to demonstrating these points, once we have developed a likelihood-evaluator program so we can fit a hurdle model using `ml`, we can easily extend our modeling capabilities by utilizing `ml`'s `cluster()`, `svy`, and `constraints()` options. Below is a likelihood-evaluator program capable of fitting a Poisson hurdle model that is equivalent to the two-step model we presented above.

```

program hurdle_ll
  version 8
  args lnf beta1 beta2
  tempvar pi lambda
  quietly generate double 'pi' = exp('beta1')
  quietly generate double 'lambda' = exp('beta2')
  quietly replace 'lnf' = cond($ML_y1==0, -'pi', ///
    log(1-exp(-'pi')) + $ML_y1*'beta2' - ///
    log(exp('lambda')-1) - lngamma($ML_y1+1))
end

```

Using the same simulated data as before, we invoke the `ml` estimator with the commands

```

. ml model lf hurdle_ll (y = x1 x2) (x1 x2)
. ml max, nolog

```

```

Log likelihood = -1361.543
Number of obs   =      1000
Wald chi2(2)    =      210.26
Prob > chi2     =      0.0000

```

	y	Coef.	Std. Err.	z	P> z	[95% Conf. Interval]	
eq1							
	x1	.9287196	.0647568	14.34	0.000	.8017986	1.055641
	x2	.0998809	.0497777	2.01	0.045	.0023184	.1974435
	_cons	.1495369	.0491899	3.04	0.002	.0531264	.2459473
eq2							
	x1	.2472253	.0363057	6.81	0.000	.1760675	.3183831
	x2	.6801573	.0319771	21.27	0.000	.6174832	.7428313
	_cons	.0166541	.0528191	0.32	0.753	-.0868694	.1201777

Comparing the results of the `ml` estimator with the results of `glm` and `trpois0`, we see that the parameter estimates and their standard errors are the same. Also, the log likelihood we obtained using the `ml` estimator is equal to the sum of the log likelihoods from the two-step estimation.

If the data are clustered, then observations within any cluster cannot be assumed to be independent, and the estimated variances of the model's parameters can be biased. We can obtain unbiased estimates in the presence of clustering simply by adding the `cluster(clusterid)` option to the `ml model` statement

```
. ml model lf hurdle_ll (xp = x1 x2) (x1 x2), cluster(clusterid)
```

where *clusterid* is a variable that identifies the clusters.

If your data were collected through a complex survey design, you must account for the sampling design to obtain unbiased variance estimates for the population parameters that are being estimated and, if the sample is weighted, to obtain unbiased estimates of the population parameters themselves. To account for the complex survey design, just `svyset` your data and add the `svy` option to the `ml model` statement

```
. svyset [pweight=weightvar], strata(strataid) psu(psuid) fpc(fpcvar)
. ml model lf hurdle_ll (xp = x1 x2) (x1 x2), svy
```

where *weightvar* is a variable containing the sampling weights, *strataid* is a variable that identifies the strata, *psuid* is a variable that identifies the psu, and *fpcvar* is a variable containing a finite population correction.

To perform constrained estimation, specify a set of numbered constraints using the `constraint` command and add the `constraints(numlist)` option to the `ml model` statement.

5 Summary

This article has presented a general form of the likelihood for hurdle models. Because the likelihood function is separable, with respect to the parameters to be estimated, we have shown that hurdle models can be represented as the sum of two independent models: a binomial probability model and a truncated-at-zero count model. Also, the parameters of a hurdle model can be estimated by fitting the two component models separately. This feature of hurdle models sets them apart from popular extensions to hurdle models such as the zero-inflated Poisson (**zip**) and the zero-inflated negative binomial (**zinb**) models. The **zip** and **zinb** models allow for a mixing process for the zeros, so the likelihoods are not separable with respect to the parameters to be estimated. An `ml` estimator, although unnecessary for estimation, has been provided for a hurdle model with a complementary log-log binomial probability model and a truncated-at-zero Poisson model to demonstrate the relative ease with which we can extend Stata's modeling capabilities to include options for dealing with clustered data, complex survey data, and constraints. While it is possible to write an all-encompassing command for fitting hurdle models in Stata, given the wide variety of possible models and the ability to fit the components of the hurdle model separately, the utility of doing so seems negligible.

6 References

- Cameron, A. C. and P. K. Trivedi. 1998. *Regression Analysis of Count Data*. New York: Cambridge University Press.
- Hilbe, J. 1999. sg102: Zero-truncated Poisson and negative binomial regression. *Stata Technical Bulletin* 47: 37–40. In *Stata Technical Bulletin Reprints*, vol. 8, 233–236. College Station, TX: Stata Press.
- Hilbe, J. and W. Linde-Zwirble. 1998. sg44.1: Correction to random number generators. *Stata Technical Bulletin* 41: 23. In *Stata Technical Bulletin Reprints*, vol. 7, 166. College Station, TX: Stata Press.
- Mullahy, J. 1986. Specification and testing of some modified count data models. *Journal of Econometrics* 3: 341–365.

About the Author

Allen McDowell is Director of Technical Services at Stata Corporation.