zip — Zero-inflated Poisson regression

Description | Quick start | Menu | Syntax
Options | Remarks and examples | Stored results | Methods and formulas
References | Also see

Description

`zip` fits a zero-inflated Poisson (ZIP) model to count data with excess zero counts. The ZIP model assumes that the excess zero counts come from a logit or probit model and the remaining counts come from a Poisson model.

Quick start

Zero-inflated Poisson model of y on `x1` and `x2` with inflation modeled using `x3`

```
zip y x1 x2, inflate(x3)
```

Use a probit model instead of a logit model to predict excess zeros

```
zip y x1 x2, inflate(x3) probit
```

Menu

Statistics > Count outcomes > Zero-inflated Poisson regression

Syntax

```
zip depvar [indepvars] [if] [in] [weight] ,
    inflate(varlist[ , offset(varname) ] | _cons) [options]
```

options	Description
Model	
* `inflate( )`	equation that determines whether the count is zero
`noconstant`	suppress constant term
`exposure(`*varname*$_e$`)`	include ln(*varname*$_e$) in model with coefficient constrained to 1
`offset(`*varname*$_o$`)`	include *varname*$_o$ in model with coefficient constrained to 1
`constraints(`*constraints*`)`	apply specified linear constraints
`probit`	use probit model to characterize excess zeros; default is logit
SE/Robust	
`vce(`*vcetype*`)`	*vcetype* may be `oim`, `robust`, `cluster` *clustvar*, `opg`, `bootstrap`, or `jackknife`
Reporting	
`level(`#`)`	set confidence level; default is `level(95)`
`irr`	report incidence-rate ratios
`nocnsreport`	do not display constraints
display_options	control columns and column formats, row spacing, line width, display of omitted variables and base and empty cells, and factor-variable labeling
Maximization	
maximize_options	control the maximization process; seldom used
`collinear`	keep collinear variables
`coeflegend`	display legend instead of statistics

*`inflate(`*varlist*`[ , offset(`*varname*`) ] | _cons)` is required.

indepvars and *varlist* may contain factor variables; see **[U] 11.4.3 Factor variables**.

`bayes`, `bayesboot`, `bootstrap`, `by`, `collect`, `fp`, `jackknife`, `rolling`, `statsby`, and `svy` are allowed; see **[U] 11.1.10 Prefix commands**. For more details, see [BAYES] **bayes: zip**.

Weights are not allowed with the `bootstrap` prefix; see [R] **bootstrap**.

`vce()` and weights are not allowed with the `svy` prefix; see [SVY] **svy**.

`fweights`, `iweights`, and `pweights` are allowed; see **[U] 11.1.6 weight**.

`collinear` and `coeflegend` do not appear in the dialog box.

See **[U] 20 Estimation and postestimation commands** for more capabilities of estimation commands.

Options

Model

`inflate(`*varlist*`[ , offset(`*varname*`)] | _cons)` specifies the equation that determines whether the observed count is zero. Conceptually, omitting `inflate()` would be equivalent to fitting the model with `poisson`; see [R] **poisson**.

`inflate(`*varlist*`[, offset(`*varname*`)])` specifies the variables in the equation. You may optionally include an offset for this *varlist*.

`inflate(_cons)` specifies that the equation determining whether the count is zero contains only an intercept. To run a zero-inflated model of *depvar* with only an intercept in both equations, type `zip` *depvar*`, inflate(_cons)`.

`noconstant`, `exposure(`*varname*$_e$`)`, `offset(`*varname*$_o$`)`, `constraints(`*constraints*`)`; see [R] **Estimation options**.

`probit` requests that a probit, instead of logit, model be used to characterize the excess zeros in the data.

SE/Robust

`vce(`*vcetype*`)` specifies the type of standard error reported, which includes types that are derived from asymptotic theory (`oim`, `opg`), that are robust to some kinds of misspecification (`robust`), that allow for intragroup correlation (`cluster` *clustvar*), and that use bootstrap or jackknife methods (`bootstrap`, `jackknife`); see [R] ***vce_option***.

Reporting

`level(`#`)`; see [R] **Estimation options**.

`irr` reports estimated coefficients transformed to incidence-rate ratios, that is, e^b rather than b. Standard errors and confidence intervals are similarly transformed. This option affects how results are displayed, not how they are estimated or stored. `irr` may be specified at estimation or when replaying previously estimated results.

`nocnsreport`; see [R] **Estimation options**.

display_options: `noci`, `nopvalues`, `noomitted`, `vsquish`, `noemptycells`, `baselevels`, `allbaselevels`, `nofvlabel`, `fvwrap(`#`)`, `fvwrapon(`*style*`)`, `cformat(`%*fmt*`)`, `pformat(`%*fmt*`)`, `sformat(`%*fmt*`)`, and `nolstretch`; see [R] **Estimation options**.

Maximization

maximize_options: `difficult`, `technique(`*algorithm_spec*`)`, `iterate(`#`)`, `[no]log`, `trace`, `gradient`, `showstep`, `hessian`, `showtolerance`, `tolerance(`#`)`, `ltolerance(`#`)`, `nrtolerance(`#`)`, `nonrtolerance`, and `from(`*init_specs*`)`; see [R] **Maximize**. These options are seldom used.

Setting the optimization type to `technique(bhhh)` resets the default *vcetype* to `vce(opg)`.

The following options are available with `zip` but are not shown in the dialog box:

`collinear`, `coeflegend`; see [R] **Estimation options**.

Remarks and examples

Zero-inflated Poisson (ZIP) models address the case when the data contain a higher fraction of zeros than is likely to be generated from a Poisson model. Having a large proportion of zero observations, in itself, does not necessarily mean that we have the excess zeros problem. For instance, a Poisson model with a mean value of 0.2 predicts that $P(Y = 0) = \exp(-0.2) \approx 0.82$. However, the range of possible outcomes is restricted because of the small variance, 0.2 (mean and variance are equal for a Poisson model), as shown by $P(Y > 3) \approx 0.00006$. Unlike Poisson models, ZIP models allow us to have a large fraction of zeros without restricting the range of outcomes.

ZIP models assume that an observation is 0 with a probability p or is a realization of a Poisson random variable, which can also be 0, with a probability $1 - p$. For instance, you might count how many fish each visitor to a park catches. Many visitors may catch zero, because they do not fish (as opposed to being unsuccessful). Using a logit or probit model, you may model the probability p of whether a person does not fish depending on several covariates related to fishing. Using a Poisson distribution, you may model how many fish a person catches depending on several covariates having to do with the success of catching fish (type of lure or bait, time of day, temperature, season, etc.). This is the type of data for which the `zip` command is useful.

See Long (1997, 242–247) and Cameron and Trivedi (2005, 680–681) for a discussion of the ZIP model and other zero-modified count models.

▷ Example 1: Fitting a ZIP model

We have fictional data on the number of fish caught (`count`) by visitors to a national park on a particular day. Some of the visitors do not fish, but we do not have the data on whether a person fished; we merely have data on how many fish were caught together with several covariates.

Variable `count` exhibits an excess of zero observations (142 of 250 observations), beyond what would be expected from a Poisson model. We suspect that the number of zeros may be inflated because many visitors are not fishing. That is, a zero observation may be the result of a visitor who was unfortunate and caught no fish but may also be because the visitor did not fish. A standard Poisson model (see [R] **poisson**) treats these two types of zero observations as a homogeneous group, which typically leads to biased statistical results. We would like to distinguish between the two types of zeros and possibly draw inference for them separately (see example 1 in [R] **zip postestimation**).

The `zip` command allows us to model the two types of zeros. First, using the required the `inflate()` option, we model whether a visitor fishes as a function of the number of children accompanying him or her (`child`) and whether he or she is camping (`camper`). Next, we assume the response variable, `count`, depends on whether the visitor used a live bait (`livebait`) and the number of persons (`persons`), which includes the visitor and any adults or children accompanying him. Note that `persons` is always greater than `child`.

```
. use https://www.stata-press.com/data/r19/fish
(Fictional fishing data)
. zip count persons livebait, inflate(child camper)
Fitting constant-only model:
Iteration 0:  Log likelihood =  -1347.807
Iteration 1:  Log likelihood = -1305.3245
Iteration 2:  Log likelihood = -1104.3005
Iteration 3:  Log likelihood = -1103.9426
Iteration 4:  Log likelihood = -1103.9425
Fitting full model:
Iteration 0:  Log likelihood = -1103.9425
Iteration 1:  Log likelihood =  -896.2346
Iteration 2:  Log likelihood = -851.61723
Iteration 3:  Log likelihood = -850.70435
Iteration 4:  Log likelihood = -850.70142
Iteration 5:  Log likelihood = -850.70142
Zero-inflated Poisson regression                  Number of obs =    250
Inflation model: logit                            Nonzero obs   =    108
                                                  Zero obs      =    142
                                                  LR chi2(2)    = 506.48
Log likelihood = -850.7014                        Prob > chi2   = 0.0000
```

count	Coefficient	Std. err.	z	P>|z|	[95% conf.	interval]
count						
persons	.8068853	.0453288	17.80	0.000	.7180424	.8957281
livebait	1.757289	.2446082	7.18	0.000	1.277866	2.236713
_cons	-2.178472	.2860289	-7.62	0.000	-2.739078	-1.617865
inflate						
child	1.602571	.2797719	5.73	0.000	1.054228	2.150913
camper	-1.015698	.365259	-2.78	0.005	-1.731593	-.2998038
_cons	-.4922872	.3114562	-1.58	0.114	-1.10273	.1181558

Coefficients in the upper half of the table correspond to the Poisson model for individuals who fished. For instance, among visitors who fished, using a live bait increases the expected number of caught fish by a factor of $\exp(1.7572) \approx 5.8$, holding other covariates constant.

◁

▷ Example 2: Comparing model fit

When you have count data, you may want to test whether a conventional count data model or a zero-inflated count data model is preferable. The classical likelihood-ratio test cannot be used here because the models are not nested. But we can use information criteria such as the AIC and BIC to check whether the standard or zero-inflated model is more appropriate.

Continuing with our fishing example, let's check whether the standard Poisson or ZIP model is more appropriate for our data. First, we store the estimation results from the previous ZIP model by typing

```
. estimates store zip
```

Next, we fit the Poisson model corresponding to the main equation of the ZIP model and store its results as pois:

```
. poisson count persons livebait
  (output omitted)
. estimates store pois
```

We use estimates stats to display the AIC and BIC values for the two models.

```
. estimates stats pois zip
Akaike's information criterion and Bayesian information criterion
```

Model	N	ll(null)	ll(model)	df	AIC	BIC
pois	250	-1647.716	-1312.178	3	2630.356	2640.92
zip	250	-1103.942	-850.7014	6	1713.403	1734.532

Note: BIC uses N = number of observations. See [R] IC note.

The ZIP model has smaller AIC and BIC values; we thus conclude that it fits our data better than the standard Poisson model.

◁

Stored results

zip stores the following in e():

Scalars

e(N)	number of observations
e(N_zero)	number of zero observations
e(k)	number of parameters
e(k_eq)	number of equations in e(b)
e(k_eq_model)	number of equations in overall model test
e(k_dv)	number of dependent variables
e(df_m)	model degrees of freedom
e(ll)	log likelihood
e(ll_0)	log likelihood, constant-only model
e(N_clust)	number of clusters
e(chi2)	χ^2
e(p)	*p*-value for model test
e(rank)	rank of e(V)
e(ic)	number of iterations
e(rc)	return code
e(converged)	1 if converged, 0 otherwise

Macros

e(cmd)	zip
e(cmdline)	command as typed
e(depvar)	name of dependent variable
e(inflate)	logit or probit
e(wtype)	weight type
e(wexp)	weight expression
e(title)	title in estimation output
e(clustvar)	name of cluster variable
e(offset1)	offset
e(offset2)	offset for inflate()
e(chi2type)	Wald or LR; type of model χ^2 test
e(vce)	*vcetype* specified in vce()
e(vcetype)	title used to label Std. err.

e(opt)	type of optimization
e(which)	max or min; whether optimizer is to perform maximization or minimization
e(ml_method)	type of ml method
e(user)	name of likelihood-evaluator program
e(technique)	maximization technique
e(properties)	b V
e(predict)	program used to implement predict
e(asbalanced)	factor variables fvset as asbalanced
e(asobserved)	factor variables fvset as asobserved

Matrices

e(b)	coefficient vector
e(Cns)	constraints matrix
e(ilog)	iteration log (up to 20 iterations)
e(gradient)	gradient vector
e(V)	variance–covariance matrix of the estimators
e(V_modelbased)	model-based variance

Functions

e(sample)	marks estimation sample

In addition to the above, the following is stored in r():

Matrices

r(table)	matrix containing the coefficients with their standard errors, test statistics, p-values, and confidence intervals

Note that results stored in r() are updated when the command is replayed and will be replaced when any r-class command is run after the estimation command.

Methods and formulas

Consider the formulation of a zero-inflated model as presented in Lambert (1992). Define

$$\begin{aligned}\xi_j^{\beta} &= \mathbf{x}_j\boldsymbol{\beta} + \text{offset}_j^{\beta}\\ \xi_j^{\gamma} &= \mathbf{z}_j\boldsymbol{\gamma} + \text{offset}_j^{\gamma}\\ \lambda_j &= \exp\left(\xi_j^{\beta}\right)\\ F_j &= F\left(\xi_j^{\gamma}\right)\end{aligned}$$

where $F(\cdot)$ is the inverse of the logit function or, if the probit option was specified, the inverse of the probit function (or the standard normal cumulative distribution function). All subjects are assumed to be independent with the jth response determined as follows:

$$\begin{aligned}Y_j &= 0 \qquad \text{with probability } F_j\\ Y_j &\sim \text{Poisson}\left(\lambda_j\right) \text{ with probability } 1 - F_j\end{aligned}$$

In other words,

$$\begin{aligned}\Pr(Y_j = 0|\mathbf{x}_j, \mathbf{z}_j) &= F_j + \left(1 - F_j\right)\exp\left(-\lambda_j\right)\\ \Pr(Y_j = n|\mathbf{x}_j, \mathbf{z}_j) &= \left(1 - F_j\right)\exp\left(-\lambda_j\right)\frac{\lambda_j^n}{n!} \qquad \text{for } n = 1, 2, \ldots\end{aligned}$$

The zip command maximizes the log-likelihood $\ln L$, defined by

$$\ln L = \sum_{j \in S} w_j \ln\left\{F_j + (1 - F_j)\exp(-\lambda_j)\right\} + \sum_{j \notin S} w_j \left\{\ln(1 - F_j) - \lambda_j + \xi_j y_j - \ln(y_j!)\right\}$$

where w_j are the weights and S is the set of observations for which the observed outcome $y_j = 0$.

This command supports the Huber/White/sandwich estimator of the variance and its clustered version using vce(robust) and vce(cluster *clustvar*), respectively. See [P] **_robust**, particularly *Maximum likelihood estimators* and *Methods and formulas*.

zip also supports estimation with survey data. For details on VCEs with survey data, see [SVY] **Variance estimation**.

References

Cameron, A. C., and P. K. Trivedi. 2005. *Microeconometrics: Methods and Applications*. New York: Cambridge University Press.

Cummings, T. H., and J. W. Hardin. 2019. Modeling count data with marginalized zero-inflated distributions. *Stata Journal* 19: 499–509.

Desmarais, B. A., and J. J. Harden. 2013. Testing for zero inflation in count models: Bias correction for the Vuong test. *Stata Journal* 13: 810–835.

Lambert, D. 1992. Zero-inflated Poisson regression, with an application to defects in manufacturing. *Technometrics* 34: 1–14. https://doi.org/10.2307/1269547.

Long, J. S. 1997. *Regression Models for Categorical and Limited Dependent Variables*. Thousand Oaks, CA: Sage.

Long, J. S., and J. Freese. 2001. Predicted probabilities for count models. *Stata Journal* 1: 51–57.

———. 2014. *Regression Models for Categorical Dependent Variables Using Stata*. 3rd ed. College Station, TX: Stata Press.

Mullahy, J. 1986. Specification and testing of some modified count data models. *Journal of Econometrics* 33: 341–365. https://doi.org/10.1016/0304-4076(86)90002-3.

Xia, Y., Y. Zhou, and T. Cai. 2019. gidm: A command for generalized inflated discrete models. *Stata Journal* 19: 698–718.

Also see

[R] **zip postestimation** — Postestimation tools for zip

[R] **zinb** — Zero-inflated negative binomial regression

[R] **nbreg** — Negative binomial regression

[R] **poisson** — Poisson regression

[R] **tnbreg** — Truncated negative binomial regression

[R] **tpoisson** — Truncated Poisson regression

[BAYES] **bayes: zip** — Bayesian zero-inflated Poisson regression

[SVY] **svy estimation** — Estimation commands for survey data

[XT] **xtpoisson** — Fixed-effects, random-effects, and population-averaged Poisson models

[U] **20 Estimation and postestimation commands**