zinb — Zero-inflated negative binomial regression

Description

`zinb` fits a zero-inflated negative binomial (ZINB) model to overdispersed count data with excess zero counts. The ZINB model assumes that the excess zero counts come from a logit or probit model and the remaining counts come from a negative binomial model.

Quick start

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

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

And conduct likelihood-ratio test against ZIP model

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

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

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

Menu

Statistics > Count outcomes > Zero-inflated negative binomial regression

Syntax

```
zinb 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
`zip`	perform ZIP likelihood-ratio test
`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: zinb**.

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

`vce()`, `zip`, 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 `nbreg`.

`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 `zinb` *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^{β_i} rather than β_i. 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.

`zip` requests that a likelihood-ratio test comparing the ZINB model with the zero-inflated Poisson model be included in the output.

`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 `zinb` but are not shown in the dialog box:

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

Remarks and examples

Zero-inflated negative binomial (ZINB) models are used to model count data that have a higher fraction of zeros than is likely to be generated by a standard negative binomial model. To account for excess zeros, ZINB models assume that these excess zeros come from a model other than the negative binomial model. A zero that comes from this other model is known as a "degenerate zero".

The negative binomial overdispersion parameter, α, differentiates the ZINB model from the zero-inflated Poisson (ZIP) model (see [R] **zip**). Here overdispersion refers to the fact that the negative binomial variance is greater than its mean, whereas the Poisson variance is equal to its mean. Thus, values of $\alpha > 1$ indicate overdispersion. The larger the α, the greater the negative binomial variance. See *Methods and formulas* in [R] **nbreg** for further discussion of negative binomial overdispersion.

The `zinb` command fits ZINB models and provides two choices for modeling the excess zeros: the default logit function or, when the `probit` option is specified, the probit function. Both functions are symmetric about zero, but the logistic function has more area under the tails.

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

▷ Example 1: Fitting a ZINB model

In example 1 of [R] **zip**, we fit a zero-inflated Poisson model using the `zip` command to the fictional data on the number of fish caught by visitors to a national park. Let's fit a ZINB model to these data.

Just like with `zip`, we use the required the `inflate()` option to model whether a visitor fishes as a function of the number of accompanying children (`child`) and whether the visitor 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 in the party (`persons`), which includes the visitor plus other adults and children.

```
. use https://www.stata-press.com/data/r19/fish
(Fictional fishing data)
. zinb count persons livebait, inflate(child camper)
Fitting constant-only model:
Iteration 0:  Log likelihood = -519.33992
Iteration 1:  Log likelihood = -451.38662
Iteration 2:  Log likelihood = -444.49118
Iteration 3:  Log likelihood = -442.96272
Iteration 4:  Log likelihood = -442.71065
Iteration 5:  Log likelihood = -442.66718
Iteration 6:  Log likelihood =  -442.6631
Iteration 7:  Log likelihood = -442.66299
Iteration 8:  Log likelihood = -442.66299
Fitting full model:
Iteration 0:  Log likelihood = -442.66299  (not concave)
Iteration 1:  Log likelihood = -432.83107  (not concave)
Iteration 2:  Log likelihood = -426.32934
Iteration 3:  Log likelihood = -413.75019
Iteration 4:  Log likelihood = -403.09586
Iteration 5:  Log likelihood = -401.56013
Iteration 6:  Log likelihood = -401.54781
Iteration 7:  Log likelihood = -401.54776
Iteration 8:  Log likelihood = -401.54776
Zero-inflated negative binomial regression              Number of obs =    250
Inflation model: logit                                  Nonzero obs   =    108
                                                        Zero obs      =    142
                                                        LR chi2(2)    =  82.23
Log likelihood = -401.5478                              Prob > chi2   = 0.0000
```

count	Coefficient	Std. err.	z	P>|z|	[95% conf.	interval]
count						
persons	.9742984	.1034938	9.41	0.000	.7714543	1.177142
livebait	1.557523	.4124424	3.78	0.000	.7491503	2.365895
_cons	-2.730064	.476953	-5.72	0.000	-3.664874	-1.795253
inflate						
child	3.185999	.7468551	4.27	0.000	1.72219	4.649808
camper	-2.020951	.872054	-2.32	0.020	-3.730146	-.3117567
_cons	-2.695385	.8929071	-3.02	0.003	-4.44545	-.9453189
/lnalpha	.5110429	.1816816	2.81	0.005	.1549535	.8671323
alpha	1.667029	.3028685			1.167604	2.380076

The coefficients in the first equation of the coefficient table, labeled `count`, correspond to the negative binomial 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 $\exp(1.5575) \approx 4.7$, holding other covariates constant.

The confidence interval for `alpha` indicates that the ZINB model is more appropriate than the ZIP model. To confirm this, you can run `zinb` and specify the `zip` option to obtain the ZIP likelihood-ratio test.

The inflate equation models whether the visitor does not fish. We can use margins to obtain a better understanding of how the inflate equation affects the occurrence of the excess zero counts. We specify margins's options dydx(child camper) and predict(pr). pr is predict's option for estimating the probability of a degenerate zero or, in our example, the probability of not fishing; see the *margins* section in [R] **zinb postestimation**.

```
. margins, dydx(child camper) predict(pr)
Average marginal effects                                  Number of obs = 250
Model VCE: OIM

Expression: Pr(count=0), predict(pr)
dy/dx wrt:  child camper
```

	dy/dx	Delta-method std. err.	z	P>|z|	[95% conf.	interval]
child	.257531	.029941	8.60	0.000	.1988477	.3162144
camper	-.1633578	.0503938	-3.24	0.001	-.2621277	-.0645878

The margins output tells us that a visitor is less likely to be visiting the park to fish if accompanied by children and more likely to fish if camping.

You also may want to evaluate whether a standard negative binomial model is adequate to fit the data. This can be done using information criteria; see example 2 in [R] **zip**.

◁

Stored results

zinb 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_aux)	number of auxiliary parameters
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(df_c)	degrees of freedom for comparison test
e(N_clust)	number of clusters
e(chi2)	χ^2
e(p)	p-value for model test
e(chi2_cp)	χ^2 for test of $\alpha = 0$
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)	zinb
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(chi2_cpt)	Wald or LR; type of model χ^2 test corresponding to e(chi2_cp)
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

The zinb command maximizes a likelihood function that is a mixture of the logistic (or probit) and negative binomial distributions. The logistic distribution models the unobserved process that creates the excess zeros, and the negative binomial distribution models the counts. Define

$$\begin{aligned}
\xi_j^{\beta} &= \mathbf{x}_j\boldsymbol{\beta} + \text{offset}_j^{\beta} \\
\xi_j^{\gamma} &= \mathbf{z}_j\boldsymbol{\gamma} + \text{offset}_j^{\gamma} \\
\mu_j &= \exp(\xi_j^{\beta}) \\
p_j &= 1/(1+\alpha\mu_j) \\
m &= 1/\alpha
\end{aligned}$$

Here the vector $\mathbf{x}_j$ contains the covariates specified in *indepvars* for the jth observation, and $\mathbf{z}_j$ contains the covariates specified in the inflate() option. Similarly, estimates for $\boldsymbol{\beta}$ are found in the first equation of the zinb coefficient table (labeled after *depvar*), and the estimates for $\boldsymbol{\gamma}$ are found in the second equation of the coefficient table (labeled inflate). The parameter α is the negative binomial

overdispersion parameter, and its estimate is the ancillary parameter labeled `alpha` in the coefficient table. Parameters p_j, m, and μ_j are parameters of a negative binomial distribution; see *Methods and formulas* in [R] **nbreg** for details.

The log likelihood maximized by `zinb` is

$$\begin{aligned}\ln L = &\sum_{j\in S} w_j \ln\left\{F_j + (1-F_j)p_j^m\right\} + \\ &\sum_{j\notin S} w_j\Big\{\ln(1-F_j) + \ln\Gamma(m+y_j) - \ln\Gamma(y_j+1) \\ &\qquad - \ln\Gamma(m) + m\ln p_j + y_j\ln(1-p_j)\Big\}\end{aligned}$$

where w_j are the weights, S is the set of observations for which the observed outcome $y_j = 0$, and F_j is the logistic distribution function

$$F_j = F(\xi_j^\gamma) = \exp(\xi_j^\gamma)/\{1+\exp(\xi_j^\gamma)\}$$

or, if the `probit` option is specified, the standard normal distribution function

$$F_j = F(\xi_j^\gamma) = \Phi(\xi_j^\gamma)$$

From Long (1997), the variance of the mixture distribution is

$$\mathrm{Var}(y_j|\mathbf{x}_i,\mathbf{z}_i) = \mu_j(1-F_j)\{1+\mu_j(F_j+\alpha)\}$$

When F_j is zero, we have the variance of the negative binomial distribution; when $F_j > 0$, the variance can exceed that of the negative binomial distribution.

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*.

`zinb` 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.

Harris, T., J. M. Hilbe, and J. W. Hardin. 2014. Modeling count data with generalized distributions. *Stata Journal* 14: 562–579.

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] **zinb postestimation** — Postestimation tools for zinb

[R] **zip** — Zero-inflated Poisson regression

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

[R] **poisson** — Poisson regression

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

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

[BAYES] **bayes: zinb** — Bayesian zero-inflated negative binomial regression

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

[XT] **xtnbreg** — Fixed-effects, random-effects, & population-averaged negative binomial models

[U] 20 Estimation and postestimation commands