

tnbreg — Truncated negative binomial regression[Description](#)[Quick start](#)[Menu](#)[Syntax](#)[Options](#)[Remarks and examples](#)[Stored results](#)[Methods and formulas](#)[Acknowledgment](#)[References](#)[Also see](#)

Description

`tnbreg` estimates the parameters of a truncated negative binomial model by maximum likelihood. The dependent variable *depvar* is regressed on *indepvars*, where *depvar* is a positive count variable whose values are all above the truncation point.

Quick start

Truncated negative binomial regression of *y* on *x* with truncation at 0

```
tnbreg y x
```

Report incidence-rate ratios

```
tnbreg y x, irr
```

Add categorical variable *a* using [factor variable](#) syntax

```
tnbreg y x i.a
```

Same as above, but specify a constant truncation point of 2

```
tnbreg y x i.a, ll(2)
```

With exposure variable *exp*

```
tnbreg y x i.a, exposure(exp)
```

Same as above, but specifying a variable truncation point stored in variable *min*

```
tnbreg y x i.a, exposure(exp) ll(min)
```

With cluster-robust standard errors clustering by the levels of *cvar*

```
tnbreg y x i.a, exposure(exp) ll(min) vce(cluster cvar)
```

Menu

Statistics > Count outcomes > Truncated negative binomial regression

Syntax

```
tnbreg depvar [indepvars] [if] [in] [weight] [, options]
```

<i>options</i>	Description
Model	
<code>noconstant</code>	suppress constant term
<code>ll(# <i>varname</i>)</code>	truncation point; default value is <code>ll(0)</code> , zero truncation
<code>dispersion(mean)</code>	parameterization of dispersion; the default
<code>dispersion(constant)</code>	constant dispersion for all observations
<code>exposure(<i>varname</i>_e)</code>	include $\ln(\text{varname}_e)$ in model with coefficient constrained to 1
<code>offset(<i>varname</i>_o)</code>	include <i>varname</i> _o in model with coefficient constrained to 1
<code>constraints(<i>constraints</i>)</code>	apply specified linear constraints
SE/Robust	
<code>vce(<i>vcetype</i>)</code>	<i>vcetype</i> may be <code>oim</code> , <code>robust</code> , <code>cluster <i>clustvar</i></code> , <code>opg</code> , <code>bootstrap</code> , or <code>jackknife</code>
Reporting	
<code>level(#)</code>	set confidence level; default is <code>level(95)</code>
<code>nolrtest</code>	suppress likelihood-ratio test
<code>irr</code>	report incidence-rate ratios
<code>nocnsreport</code>	do not display constraints
<code>display_options</code>	control columns and column formats, row spacing, line width, display of omitted variables and base and empty cells, and factor-variable labeling
Maximization	
<code>maximize_options</code>	control the maximization process; seldom used
<code>collinear</code>	keep collinear variables
<code>coeflegend</code>	display legend instead of statistics

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

depvar and *indepvars* may contain time-series operators; see [U] 11.4.4 Time-series varlists.

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

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

`noconstant`; see [R] Estimation options.

`ll(# | varname)` specifies the truncation point, which is a nonnegative integer. The default is zero truncation, `ll(0)`.

`dispersion(mean | constant)` specifies the parameterization of the model. `dispersion(mean)`, the default, yields a model with dispersion equal to $1 + \alpha \exp(\mathbf{x}_j\beta + \text{offset}_j)$; that is, the dispersion is a function of the expected mean: $\exp(\mathbf{x}_j\beta + \text{offset}_j)$. `dispersion(constant)` has dispersion equal to $1 + \delta$; that is, it is a constant for all observations.

`exposure(varnamee)`, `offset(varnameo)`, `constraints(constraints)`; see [R] [Estimation options](#).

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](#).

`nolrtest` suppresses fitting the Poisson model. Without this option, a comparison Poisson model is fit, and the likelihood is used in a likelihood-ratio test of the null hypothesis that the dispersion parameter is zero.

`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.

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

display_options: `nocl`, `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 `tnbreg` but are not shown in the dialog box:

`collinear`, `coeflegend`; see [R] [Estimation options](#).

Remarks and examples

[stata.com](http://www.stata.com)

Grogger and Carson (1991) showed that overdispersion causes inconsistent estimation of the mean in the truncated Poisson model. To solve this problem, they proposed using the truncated negative binomial model as an alternative. If data are truncated but do not exhibit overdispersion, the truncated Poisson model is more appropriate; see [R] [tpoisson](#). For an introduction to negative binomial regression, see Cameron and Trivedi (2005, 2022) and Long and Freese (2014). For an introduction to truncated negative binomial models, see Cameron and Trivedi (2013) and Long (1997, chap. 8).

`tnbreg` fits the mean-dispersion and the constant-dispersion parameterizations of truncated negative binomial models. These parameterizations extend those implemented in `nbreg`; see [R] `nbreg`.

▷ Example 1

We illustrate the truncated negative binomial model using the 1997 MedPar dataset (Hilbe 1999). The data are from 1,495 patients in Arizona who were assigned to a diagnostic-related group (DRG) of patients having a ventilator. Length of stay (`los`), the dependent variable, is a positive integer; it cannot have zero values. The data are truncated because there are no observations on individuals who stayed for zero days.

The objective of this example is to determine whether the length of stay was related to the binary variables: `died`, `hmo`, `type1`, `type2`, and `type3`.

The `died` variable was recorded as a 0 unless the patient died, in which case, it was recorded as a 1. The other variables also adopted this encoding. The `hmo` variable was set to 1 if the patient belonged to a health maintenance organization (HMO).

The `type1`–`type3` variables indicated the type of admission used for the patient. The `type1` variable indicated an emergency admit. The `type2` variable indicated an urgent admit—that is, the first available bed. The `type3` variable indicated an elective admission. Because `type1`–`type3` were mutually exclusive, only two of the three could be used in the truncated negative binomial regression shown below.

```
. use https://www.stata-press.com/data/r18/medpar
(Arizona ventilator data)
. tnbreg los died hmo type2-type3, vce(cluster provnum) nolog
Truncated negative binomial regression      Number of obs = 1,495
Truncation point = 0                       Wald chi2(4) = 36.01
Dispersion: mean                           Prob > chi2 = 0.0000
Log pseudolikelihood = -4737.535          Pseudo R2 = 0.0139
                                           (Std. err. adjusted for 54 clusters in provnum)
```

los	Coefficient	Robust std. err.	z	P> z	[95% conf. interval]	
died	-.2521884	.061533	-4.10	0.000	-.3727908	-.1315859
hmo	-.0754173	.0533132	-1.41	0.157	-.1799091	.0290746
type2	.2685095	.0666474	4.03	0.000	.137883	.3991359
type3	.7668101	.2183505	3.51	0.000	.338851	1.194769
_cons	2.224028	.034727	64.04	0.000	2.155964	2.292091
/lnalpha	-.630108	.0764019			-.779853	-.480363
alpha	.5325343	.0406866			.4584734	.6185588

Because observations within the same hospital (`provnum`) are likely to be correlated, we specified the `vce(cluster provnum)` option. The results show that whether the patient died in the hospital and the type of admission have significant effects on the patient’s length of stay.



▷ Example 2

To illustrate truncated negative binomial regression with more complex data than the previous example, similar data were created from 100 hospitals. Each hospital had its own way of tracking patient data. In particular, hospitals only recorded data from patients with a minimum length of stay, denoted by the variable `minstay`.

Definitions for minimum length of stay varied among hospitals, typically, from 5 to 18 days. The objective of this example is the same as before: to determine whether the length of stay, recorded in `los`, was related to the binary variables: `died`, `hmo`, `type1`, `type2`, and `type3`.

The binary variables encode the same information as in [example 1](#) above. The `minstay` variable was used to allow for varying truncation points.

```
. use https://www.stata-press.com/data/r18/medproviders
. tnbreg los died hmo type2-type3, ll(minstay) vce(cluster hospital) nolog
Truncated negative binomial regression          Number of obs = 2,144
Truncation points: minstay                    Wald chi2(4) = 15.22
Dispersion: mean                             Prob > chi2 = 0.0043
Log pseudolikelihood = -7864.0928            Pseudo R2 = 0.0007
                                           (Std. err. adjusted for 100 clusters in hospital)
```

los	Coefficient	Robust std. err.	z	P> z	[95% conf. interval]	
died	.078104	.0303598	2.57	0.010	.0185998	.1376081
hmo	-.0731132	.0368897	-1.98	0.047	-.1454158	-.0008107
type2	.0294132	.0390166	0.75	0.451	-.047058	.1058845
type3	.0626348	.0540123	1.16	0.246	-.0432273	.168497
_cons	3.014964	.0290895	103.64	0.000	2.95795	3.071978
/lnalpha	-.996512	.0828691			-1.158932	-.8340916
alpha	.3691649	.0305923			.313821	.4342688

In this analysis, two variables have a statistically significant relationship with length of stay. On average, patients who died in the hospital had longer lengths of stay ($p = 0.01$). Because the coefficient for HMO is negative, that is, $b_{HMO} = -0.073$, on average, patients who were insured by an HMO had shorter lengths of stay ($p = 0.047$). The type of admission was not statistically significant ($p > 0.05$).

Stored results

`tnbreg` stores the following in `e()`:

Scalars

<code>e(N)</code>	number of observations
<code>e(k)</code>	number of parameters
<code>e(k_aux)</code>	number of auxiliary parameters
<code>e(k_eq)</code>	number of equations in <code>e(b)</code>
<code>e(k_eq_model)</code>	number of equations in overall model test
<code>e(k_dv)</code>	number of dependent variables
<code>e(df_m)</code>	model degrees of freedom
<code>e(r2_p)</code>	pseudo- R^2
<code>e(ll)</code>	log likelihood
<code>e(ll_0)</code>	log likelihood, constant-only model
<code>e(ll_c)</code>	log likelihood, comparison model
<code>e(alpha)</code>	value of alpha
<code>e(delta)</code>	value of delta
<code>e(N_clust)</code>	number of clusters
<code>e(chi2)</code>	χ^2
<code>e(chi2_c)</code>	χ^2 for comparison test
<code>e(p)</code>	p -value for model test
<code>e(rank)</code>	rank of <code>e(V)</code>
<code>e(rank0)</code>	rank of <code>e(V)</code> for constant-only model
<code>e(ic)</code>	number of iterations
<code>e(rc)</code>	return code
<code>e(converged)</code>	1 if converged, 0 otherwise

Macros

<code>e(cmd)</code>	<code>tnbreg</code>
<code>e(cmdline)</code>	command as typed
<code>e(depvar)</code>	name of dependent variable
<code>e(llopt)</code>	contents of <code>ll()</code> , or 0 if not specified
<code>e(wtype)</code>	weight type
<code>e(wexp)</code>	weight expression
<code>e(title)</code>	title in estimation output
<code>e(clustvar)</code>	name of cluster variable
<code>e(offset)</code>	linear offset variable
<code>e(chi2type)</code>	Wald or LR; type of model χ^2 test
<code>e(chi2_ct)</code>	Wald or LR; type of model χ^2 test corresponding to <code>e(chi2_c)</code>
<code>e(dispers)</code>	mean or constant
<code>e(vce)</code>	<i>vctype</i> specified in <code>vce()</code>
<code>e(vcetype)</code>	title used to label Std. err.
<code>e(opt)</code>	type of optimization
<code>e(which)</code>	max or min; whether optimizer is to perform maximization or minimization
<code>e(ml_method)</code>	type of ml method
<code>e(user)</code>	name of likelihood-evaluator program
<code>e(technique)</code>	maximization technique
<code>e(properties)</code>	<code>b V</code>
<code>e(predict)</code>	program used to implement <code>predict</code>
<code>e(asbalanced)</code>	factor variables <code>fvset</code> as <code>asbalanced</code>
<code>e(asobserved)</code>	factor variables <code>fvset</code> as <code>asobserved</code>

Matrices

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

Functions

<code>e(sample)</code>	marks estimation sample
------------------------	-------------------------

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

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

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

Methods and formulas are presented under the following headings:

- [Mean-dispersion model](#)
- [Constant-dispersion model](#)

Mean-dispersion model

A negative binomial distribution can be regarded as a gamma mixture of Poisson random variables. The number of times an event occurs, y_j , is distributed as $\text{Poisson}(\nu_j \mu_j)$. That is, its conditional likelihood is

$$f(y_j | \nu_j) = \frac{(\nu_j \mu_j)^{y_j} e^{-\nu_j \mu_j}}{\Gamma(y_j + 1)}$$

where $\mu_j = \exp(\mathbf{x}_j \boldsymbol{\beta} + \text{offset}_j)$ and ν_j is an unobserved parameter with a $\text{Gamma}(1/\alpha, \alpha)$ density:

$$g(\nu) = \frac{\nu^{(1-\alpha)/\alpha} e^{-\nu/\alpha}}{\alpha^{1/\alpha} \Gamma(1/\alpha)}$$

This gamma distribution has a mean of 1 and a variance of α , where α is our ancillary parameter.

The unconditional likelihood for the j th observation is therefore

$$f(y_j) = \int_0^\infty f(y_j | \nu) g(\nu) d\nu = \frac{\Gamma(m + y_j)}{\Gamma(y_j + 1) \Gamma(m)} p_j^m (1 - p_j)^{y_j}$$

where $p_j = 1/(1 + \alpha \mu_j)$ and $m = 1/\alpha$. Solutions for α are handled by searching for $\ln \alpha$ because α must be greater than zero. The conditional probability of observing y_j events given that y_j is greater than the truncation point τ_j is

$$\Pr(Y = y_j | y_j > \tau_j, \mathbf{x}_j) = \frac{f(y_j)}{\Pr(Y > \tau_j | \mathbf{x}_j)}$$

The log likelihood (with weights w_j and offsets) is given by

$$\begin{aligned} m = 1/\alpha \quad p_j = 1/(1 + \alpha \mu_j) \quad \mu_j = \exp(\mathbf{x}_j \boldsymbol{\beta} + \text{offset}_j) \\ \ln L = \sum_{j=1}^n w_j \left[\ln\{\Gamma(m + y_j)\} - \ln\{\Gamma(y_j + 1)\} \right. \\ \left. - \ln\{\Gamma(m)\} + m \ln(p_j) + y_j \ln(1 - p_j) - \ln\{\Pr(Y > \tau_j | p_j, m)\} \right] \end{aligned}$$

Constant-dispersion model

The constant-dispersion model assumes that y_j is conditionally distributed as $\text{Poisson}(\mu_j^*)$, where $\mu_j^* \sim \text{Gamma}(\mu_j/\delta, \delta)$ for some dispersion parameter δ [by contrast, the mean-dispersion model assumes that $\mu_j^* \sim \text{Gamma}(1/\alpha, \alpha\mu_j)$]. The log likelihood is given by

$$m_j = \mu_j/\delta \quad p = 1/(1 + \delta)$$

$$\ln L = \sum_{j=1}^n w_j \left[\ln\{\Gamma(m_j + y_j)\} - \ln\{\Gamma(y_j + 1)\} \right. \\ \left. - \ln\{\Gamma(m_j)\} + m_j \ln(p) + y_j \ln(1 - p) - \ln\{\Pr(Y > \tau_j | p, m_j)\} \right]$$

with everything else defined as shown above in the calculations for the mean-dispersion model.

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*.

`tnbreg` also supports estimation with survey data. For details on VCEs with survey data, see [SVY] [Variance estimation](#).

Acknowledgment

We gratefully acknowledge the previous work by Joseph Hilbe (1944–2017) (1999), coauthor of the Stata Press book *Generalized Linear Models and Extensions*.

References

- Cameron, A. C., and P. K. Trivedi. 2005. *Microeconometrics: Methods and Applications*. New York: Cambridge University Press.
- . 2013. *Regression Analysis of Count Data*. 2nd ed. New York: Cambridge University Press.
- . 2022. *Microeconometrics Using Stata*. 2nd ed. College Station, TX: Stata Press.
- Grogger, J. T., and R. T. Carson. 1991. Models for truncated counts. *Journal of Applied Econometrics* 6: 225–238. <https://doi.org/10.1002/jae.3950060302>.
- Hardin, J. W., and J. M. Hilbe. 2015. Regression models for count data from truncated distributions. *Stata Journal* 15: 226–246.
- Hilbe, J. M. 1999. `sg102: Zero-truncated Poisson and negative binomial regression`. *Stata Technical Bulletin* 47: 37–40. Reprinted in *Stata Technical Bulletin Reprints*, vol. 8, pp. 233–236. College Station, TX: Stata Press.
- Long, J. S. 1997. *Regression Models for Categorical and Limited Dependent Variables*. Thousand Oaks, CA: Sage.
- Long, J. S., and J. Freese. 2014. *Regression Models for Categorical Dependent Variables Using Stata*. 3rd ed. College Station, TX: Stata Press.
- Simonoff, J. S. 2003. *Analyzing Categorical Data*. New York: Springer.

Also see

[R] [tnbreg postestimation](#) — Postestimation tools for tnbreg

[R] [nbreg](#) — Negative binomial regression

[R] [poisson](#) — Poisson regression

[R] [tpoisson](#) — Truncated Poisson regression

[R] [zinb](#) — Zero-inflated negative binomial regression

[R] [zip](#) — Zero-inflated Poisson regression

[BAYES] [bayes: tnbreg](#) — Bayesian truncated 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](#)

Stata, Stata Press, and Mata are registered trademarks of StataCorp LLC. Stata and Stata Press are registered trademarks with the World Intellectual Property Organization of the United Nations. Other brand and product names are registered trademarks or trademarks of their respective companies. Copyright © 1985–2023 StataCorp LLC, College Station, TX, USA. All rights reserved.

