

cloglog — Complementary log–log regression[Description](#)[Quick start](#)[Menu](#)[Syntax](#)[Options](#)[Remarks and examples](#)[Stored results](#)[Methods and formulas](#)[Acknowledgment](#)[References](#)[Also see](#)

Description

`cloglog` fits a complementary log–log model for a binary dependent variable, typically with one of the outcomes rare relative to the other. It can also be used to fit a gompit model. `cloglog` can compute robust and cluster–robust standard errors and adjust results for complex survey designs.

Quick start

Complementary log–log model of y on x_1 and x_2

```
cloglog y x1 x2
```

With robust standard errors

```
cloglog y x1 x2, vce(robust)
```

Adjust for complex survey design using `svyset` data

```
svy: cloglog y x1 x2
```

Menu

Statistics > Binary outcomes > Complementary log–log regression

Syntax

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

<i>options</i>	Description
Model	
<code>noconstant</code>	suppress constant term
<code>offset(<i>varname</i>)</code>	include <i>varname</i> in model with coefficient constrained to 1
<code>asis</code>	retain perfect predictor variables
<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>eform</code>	report exponentiated coefficients
<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`, `fmm`, `fp`, `jackknife`, `mi estimate`, `nestreg`, `rolling`, `statsby`, `stepwise`, and `svy` are allowed; see [U] 11.1.10 **Prefix commands**. For more details, see [BAYES] `bayes: cloglog` and [FMM] `fmm: cloglog`.

`vce(bootstrap)` and `vce(jackknife)` are not allowed with the `mi estimate` prefix; see [MI] `mi estimate`.

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`, `offset(varname)`; see [R] **Estimation options**.

`asis` forces retention of perfect predictor variables and their associated perfectly predicted observations and may produce instabilities in maximization; see [R] `probit`.

`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](#).

`eform` displays the exponentiated coefficients and corresponding standard errors and confidence intervals.

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

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

Remarks and examples

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

Remarks are presented under the following headings:

[Introduction to complementary log–log regression](#)
[Robust standard errors](#)

Introduction to complementary log–log regression

`cloglog` fits maximum likelihood models with dichotomous dependent variables coded as 0/1 (or, more precisely, coded as 0 and not 0).

► Example 1

We have data on the make, weight, and mileage rating of 22 foreign and 52 domestic automobiles. We wish to fit a model explaining whether a car is foreign based on its weight and mileage. Here is an overview of our data:

```
. use https://www.stata-press.com/data/r18/auto
(1978 automobile data)
. keep make mpg weight foreign
. describe
Contains data from https://www.stata-press.com/data/r18/auto.dta
Observations:      74                1978 automobile data
Variables:         4                 13 Apr 2022 17:45
                                   (_dta has notes)
```

Variable name	Storage type	Display format	Value label	Variable label
make	str18	%-18s		Make and model
mpg	int	%8.0g		Mileage (mpg)
weight	int	%8.0gc		Weight (lbs.)
foreign	byte	%8.0g	origin	Car origin

Sorted by: foreign

Note: Dataset has changed since last saved.

```
. inspect foreign
```

```
foreign: Car origin
```

		Number of observations		
		Total	Integers	Nonintegers
#	Negative	-	-	-
#	Zero	52	52	-
#	Positive	22	22	-
#				
# #	Total	74	74	-
# #	Missing	-		
0		74		
1				

(2 unique values)

foreign is labeled and all values are documented in the label.

The variable `foreign` takes on two unique values, 0 and 1. The value 0 denotes a domestic car, and 1 denotes a foreign car.

The model that we wish to fit is

$$\Pr(\text{foreign} = 1) = F(\beta_0 + \beta_1 \text{weight} + \beta_2 \text{mpg})$$

where $F(z) = 1 - \exp\{-\exp(z)\}$.

To fit this model, we type

```
. cloglog foreign weight mpg
Iteration 0: Log likelihood = -34.054593
Iteration 1: Log likelihood = -27.869915
Iteration 2: Log likelihood = -27.742997
Iteration 3: Log likelihood = -27.742769
Iteration 4: Log likelihood = -27.742769
Complementary log-log regression          Number of obs   =       74
                                          Zero outcomes  =       52
                                          Nonzero outcomes =       22
                                          LR chi2(2)     =      34.58
                                          Prob > chi2    =      0.0000

Log likelihood = -27.742769
```

foreign	Coefficient	Std. err.	z	P> z	[95% conf. interval]	
weight	-.0029153	.0006974	-4.18	0.000	-.0042823	-.0015483
mpg	-.1422911	.076387	-1.86	0.062	-.2920069	.0074247
_cons	10.09694	3.351841	3.01	0.003	3.527448	16.66642

We find that heavier cars are less likely to be foreign and that cars yielding better gas mileage are also less likely to be foreign, at least when holding the weight of the car constant.

See [\[R\] Maximize](#) for an explanation of the output.

◀

□ Technical note

Stata interprets a value of 0 as a negative outcome (failure) and treats all other values (except missing) as positive outcomes (successes). Thus, if your dependent variable takes on the values 0 and 1, 0 is interpreted as failure and 1 as success. If your dependent variable takes on the values 0, 1, and 2, 0 is still interpreted as failure, but both 1 and 2 are treated as successes.

If you prefer a more formal mathematical statement, when you type `cloglog y x`, Stata fits the model

$$\Pr(y_j \neq 0 \mid \mathbf{x}_j) = 1 - \exp\left\{-\exp(\mathbf{x}_j\boldsymbol{\beta})\right\}$$

□

Robust standard errors

If you specify the `vce(robust)` option, `cloglog` reports robust standard errors, as described in [\[U\] 20.22 Obtaining robust variance estimates](#). For the model of `foreign` on `weight` and `mpg`, the robust calculation increases the standard error of the coefficient on `mpg` by 44%:

```

. cloglog foreign weight mpg, vce(robust)
Iteration 0: Log pseudolikelihood = -34.054593
Iteration 1: Log pseudolikelihood = -27.869915
Iteration 2: Log pseudolikelihood = -27.742997
Iteration 3: Log pseudolikelihood = -27.742769
Iteration 4: Log pseudolikelihood = -27.742769

Complementary log-log regression          Number of obs   =       74
                                          Zero outcomes  =       52
                                          Nonzero outcomes =       22
                                          Wald chi2(2)   =      29.74
                                          Prob > chi2    =      0.0000

Log pseudolikelihood = -27.742769

```

foreign	Coefficient	Robust std. err.	z	P> z	[95% conf. interval]	
weight	-.0029153	.0007484	-3.90	0.000	-.0043822	-.0014484
mpg	-.1422911	.1102466	-1.29	0.197	-.3583704	.0737882
_cons	10.09694	4.317305	2.34	0.019	1.635174	18.5587

Without `vce(robust)`, the standard error for the coefficient on `mpg` was reported to be 0.076, with a resulting confidence interval of $[-0.29, 0.01]$.

The `vce(cluster clustvar)` option can relax the independence assumption required by the complementary log–log estimator to being just independence between clusters. To demonstrate this ability, we will switch to a different dataset.

We are studying unionization of women in the United States by using the `union` dataset; see [\[XT\] xt](#). We fit the following model, ignoring that women are observed an average of 5.9 times each in this dataset:

```

. use https://www.stata-press.com/data/r18/union, clear
(NLS Women 14-24 in 1968)

. cloglog union age grade not_smsa south##c.year
Iteration 0: Log likelihood = -13606.373
Iteration 1: Log likelihood = -13540.726
Iteration 2: Log likelihood = -13540.607
Iteration 3: Log likelihood = -13540.607

Complementary log-log regression          Number of obs   =      26,200
                                          Zero outcomes  =      20,389
                                          Nonzero outcomes =       5,811
                                          LR chi2(6)    =      647.24
                                          Prob > chi2    =      0.0000

Log likelihood = -13540.607

```

union	Coefficient	Std. err.	z	P> z	[95% conf. interval]	
age	.0185346	.0043616	4.25	0.000	.009986	.0270833
grade	.0452772	.0057125	7.93	0.000	.0340809	.0564736
not_smsa	-.1886592	.0317801	-5.94	0.000	-.2509471	-.1263712
1.south	-1.422292	.3949381	-3.60	0.000	-2.196356	-.648227
year	-.0133007	.0049576	-2.68	0.007	-.0230174	-.0035839
south#c.year						
1	.0105659	.0049234	2.15	0.032	.0009161	.0202157
_cons	-1.219801	.2952374	-4.13	0.000	-1.798455	-.6411462

The reported standard errors in this model are probably meaningless. Women are observed repeatedly, and so the observations are not independent. Looking at the coefficients, we find a large southern

effect against unionization and a different time trend for the south. The `vce(cluster clustvar)` option provides a way to fit this model and obtains correct standard errors:

```
. cloglog union age grade not_smsa south##c.year, vce(cluster id) nolog
Complementary log–log regression          Number of obs   =    26,200
                                          Zero outcomes   =    20,389
                                          Nonzero outcomes =     5,811
                                          Wald chi2(6)    =    160.76
Log pseudolikelihood = -13540.607        Prob > chi2      =     0.0000
                                          (Std. err. adjusted for 4,434 clusters in idcode)
```

union	Coefficient	Robust std. err.	z	P> z	[95% conf. interval]	
age	.0185346	.0084873	2.18	0.029	.0018999	.0351694
grade	.0452772	.0125776	3.60	0.000	.0206255	.069929
not_smsa	-.1886592	.0642068	-2.94	0.003	-.3145021	-.0628162
1.south	-1.422292	.506517	-2.81	0.005	-2.415047	-.4295365
year	-.0133007	.0090628	-1.47	0.142	-.0310633	.004462
south#c.year						
1	.0105659	.0063175	1.67	0.094	-.0018162	.022948
_cons	-1.219801	.5175129	-2.36	0.018	-2.234107	-.2054942

These standard errors are larger than those reported by the inappropriate conventional calculation. By comparison, another way we could fit this model is with an equal-correlation population-averaged complementary log–log model:

```
. xtccloglog union age grade not_smsa south##c.year, pa nolog
GEE population-averaged model          Number of obs   =    26,200
Group variable: idcode                 Number of groups =    4,434
Family: Binomial                       Obs per group:
Link: Complementary log–log              min =           1
Correlation: exchangeable                avg =           5.9
                                          max =           12
                                          Wald chi2(6)    =    234.66
Scale parameter = 1                     Prob > chi2      =     0.0000
```

union	Coefficient	Std. err.	z	P> z	[95% conf. interval]	
age	.0153737	.0081156	1.89	0.058	-.0005326	.03128
grade	.0549518	.0095093	5.78	0.000	.0363139	.0735897
not_smsa	-.1045232	.0431082	-2.42	0.015	-.1890138	-.0200326
1.south	-1.714868	.3384558	-5.07	0.000	-2.378229	-1.051507
year	-.0115881	.0084125	-1.38	0.168	-.0280763	.0049001
south#c.year						
1	.0149796	.0041687	3.59	0.000	.0068091	.0231501
_cons	-1.488278	.4468005	-3.33	0.001	-2.363991	-.6125652

The coefficient estimates are similar, but these standard errors are smaller than those produced by `cloglog, vce(cluster clustvar)`. This finding is as we would expect. If the within-panel correlation assumptions are valid, the population-averaged estimator should be more efficient.

In addition to this estimator, we may use the `xtgee` command to fit a panel estimator (with complementary log–log link) and any number of assumptions on the within-idcode correlation.

`cloglog`, `vce(cluster clustvar)` is robust to assumptions about within-cluster correlation. That is, it inefficiently sums within cluster for the standard error calculation rather than attempting to exploit what might be assumed about the within-cluster correlation (as do the `xtgee` population-averaged models).

Stored results

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

Scalars

<code>e(N)</code>	number of observations
<code>e(k)</code>	number of 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(N_f)</code>	number of zero outcomes
<code>e(N_s)</code>	number of nonzero outcomes
<code>e(df_m)</code>	model degrees of freedom
<code>e(ll)</code>	log likelihood
<code>e(ll_0)</code>	log likelihood, constant-only model
<code>e(N_clust)</code>	number of clusters
<code>e(chi2)</code>	χ^2
<code>e(p)</code>	<i>p</i> -value for model test
<code>e(rank)</code>	rank of <code>e(V)</code>
<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>cloglog</code>
<code>e(cmdline)</code>	command as typed
<code>e(depvar)</code>	name of dependent variable
<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(vce)</code>	<i>vcetype</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(marginsok)</code>	predictions allowed by <code>margins</code>
<code>e(marginsnotok)</code>	predictions disallowed by <code>margins</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()`:

Matrices

<code>r(table)</code>	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

Complementary log–log analysis (related to the gompit model, so named because of its relationship to the Gompertz distribution) is an alternative to logit and probit analysis, but it is unlike these other estimators in that the transformation is not symmetric. Typically, this model is used when the positive (or negative) outcome is rare.

The log-likelihood function for complementary log–log is

$$\ln L = \sum_{j \in S} w_j \ln F(\mathbf{x}_j \mathbf{b}) + \sum_{j \notin S} w_j \ln \{1 - F(\mathbf{x}_j \mathbf{b})\}$$

where S is the set of all observations j such that $y_j \neq 0$, $F(z) = 1 - \exp\{-\exp(z)\}$, and w_j denotes the optional weights. $\ln L$ is maximized as described in [R] [Maximize](#).

We can fit a gompit model by reversing the success–failure sense of the dependent variable and using `cloglog`.

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](#). The scores are calculated as $\mathbf{u}_j = [\exp(\mathbf{x}_j \mathbf{b}) \exp\{-\exp(\mathbf{x}_j \mathbf{b})\} / F(\mathbf{x}_j \mathbf{b})] \mathbf{x}_j$ for the positive outcomes and $\{-\exp(\mathbf{x}_j \mathbf{b})\} \mathbf{x}_j$ for the negative outcomes.

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

Acknowledgment

We thank Joseph Hilbe (1944–2017), coauthor of the Stata Press book [Generalized Linear Models and Extensions](#), for providing the inspiration for the `cloglog` command.

References

- 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.
- Xu, J., and J. S. Long. 2005. Confidence intervals for predicted outcomes in regression models for categorical outcomes. *Stata Journal* 5: 537–559.

Also see

- [R] **cloglog postestimation** — Postestimation tools for cloglog
- [R] **logit** — Conditional (fixed-effects) logistic regression
- [R] **glm** — Generalized linear models
- [R] **logistic** — Logistic regression, reporting odds ratios
- [R] **scobit** — Skewed logistic regression
- [BAYES] **bayes: cloglog** — Bayesian complementary log–log regression
- [FMM] **fm: cloglog** — Finite mixtures of complementary log–log regression models
- [ME] **mecloglog** — Multilevel mixed-effects complementary log–log regression
- [MI] **Estimation** — Estimation commands for use with mi estimate
- [SVY] **svy estimation** — Estimation commands for survey data
- [XT] **xtcloglog** — Random-effects and population-averaged cloglog 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.

