Title

rreg — Robust regression

Syntax

`rreg` *depvar* [*indepvars*] [*if*] [*in*] [, *options*]

options	Description
Model	
`tune(`#`)`	use # as the biweight tuning constant; default is `tune(7)`
Reporting	
`level(`#`)`	set confidence level; default is `level(95)`
`genwt(`*newvar*`)`	create *newvar* containing the weights assigned to each observation
display_options	control column formats, row spacing, line width, display of omitted variables and base and empty cells, and factor-variable labeling
Optimization	
optimization_options	control the optimization process; seldom used
`graph`	graph weights during convergence
`coeflegend`	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**.

`by`, `mfp`, `mi estimate`, `rolling`, and `statsby` are allowed; see **[U] 11.1.10 Prefix commands**.

`coeflegend` does not appear in the dialog box.

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

Menu

Statistics > Linear models and related > Other > Robust regression

Description

`rreg` performs one version of robust regression of *depvar* on *indepvars*.

Also see *Robust standard errors* in **[R] regress** for standard regression with robust variance estimates and **[R] qreg** for quantile (including median or least-absolute-residual) regression.

Options

Model

`tune(#)` is the biweight tuning constant. The default is 7, meaning seven times the median absolute deviation (MAD) from the median residual; see *Methods and formulas*. Lower tuning constants downweight outliers rapidly but may lead to unstable estimates (less than 6 is not recommended). Higher tuning constants produce milder downweighting.

Reporting

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

`genwt(`*newvar*`)` creates the new variable *newvar* containing the weights assigned to each observation.

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

Optimization

optimization_options: `iterate(#)`, `tolerance(#)`, [`no`]`log`. `iterate()` specifies the maximum number of iterations; iterations stop when the maximum change in weights drops below `tolerance()`; and `log`/`nolog` specifies whether to show the iteration log. These options are seldom used.

`graph` allows you to graphically watch the convergence of the iterative technique. The weights obtained from the most recent round of estimation are graphed against the weights obtained from the previous round.

The following option is available with `rreg` but is not shown in the dialog box:

`coeflegend`; see [R] **estimation options**.

Remarks and examples

stata.com

`rreg` first performs an initial screening based on Cook's distance > 1 to eliminate gross outliers before calculating starting values and then performs Huber iterations followed by biweight iterations, as suggested by Li (1985).

▷ Example 1

We wish to examine the relationship between mileage rating, weight, and location of manufacture for the 74 cars in our automobile data. As a point of comparison, we begin by fitting an ordinary regression:

```
. use http://www.stata-press.com/data/r13/auto
(1978 Automobile Data)

. regress mpg weight foreign

      Source |       SS       df       MS              Number of obs =      74
-------------+------------------------------           F(  2,    71) =   69.75
       Model |   1619.2877     2  809.643849           Prob > F      =  0.0000
    Residual |  824.171761    71   11.608053           R-squared     =  0.6627
-------------+------------------------------           Adj R-squared =  0.6532
       Total |  2443.45946    73  33.4720474           Root MSE      =  3.4071

------------------------------------------------------------------------------
         mpg |      Coef.   Std. Err.      t    P>|t|     [95% Conf. Interval]
-------------+----------------------------------------------------------------
      weight |  -.0065879   .0006371   -10.34   0.000    -.0078583   -.0053175
     foreign |  -1.650029   1.075994    -1.53   0.130      -3.7955    .4954422
       _cons |    41.6797   2.165547    19.25   0.000     37.36172    45.99768
------------------------------------------------------------------------------
```

We now compare this with the results from `rreg`:

```
. rreg mpg weight foreign

   Huber iteration 1:  maximum difference in weights = .80280176
   Huber iteration 2:  maximum difference in weights = .2915438
   Huber iteration 3:  maximum difference in weights = .08911171
   Huber iteration 4:  maximum difference in weights = .02697328
Biweight iteration 5:  maximum difference in weights = .29186818
Biweight iteration 6:  maximum difference in weights = .11988101
Biweight iteration 7:  maximum difference in weights = .03315872
Biweight iteration 8:  maximum difference in weights = .00721325

Robust regression                                      Number of obs =      74
                                                       F(  2,    71) =  168.32
                                                       Prob > F      =  0.0000

------------------------------------------------------------------------------
         mpg |      Coef.   Std. Err.      t    P>|t|     [95% Conf. Interval]
-------------+----------------------------------------------------------------
      weight |  -.0063976   .0003718   -17.21   0.000     -.007139   -.0056562
     foreign |  -3.182639    .627964    -5.07   0.000    -4.434763   -1.930514
       _cons |   40.64022   1.263841    32.16   0.000      38.1202    43.16025
------------------------------------------------------------------------------
```

Note the large change in the `foreign` coefficient.

◁

❑ Technical note

It would have been better if we had fit the previous robust regression by typing `rreg mpg weight foreign, genwt(w)`. The new variable, `w`, would then contain the estimated weights. Let's pretend that we did this:

```
. rreg mpg weight foreign, genwt(w)
  (output omitted )
. summarize w, detail
                    Robust Regression Weight
─────────────────────────────────────────────────────────────
      Percentiles      Smallest
 1%            0              0
 5%     .0442957              0
10%     .4674935              0       Obs                  74
25%     .8894815       .0442957       Sum of Wgt.          74

50%     .9690193                      Mean           .8509966
                        Largest       Std. Dev.      .2746451
75%     .9949395       .9996715
90%     .9989245       .9996953       Variance       .0754299
95%     .9996715       .9997343       Skewness      -2.287952
99%     .9998585       .9998585       Kurtosis       6.874605
```

We discover that 3 observations in our data were dropped altogether (they have weight 0). We could further explore our data:

```
. sort w

. list make mpg weight w if w <.467, sep(0)

     ┌─────────────────────────────────────────────┐
     │ make              mpg   weight            w │
     ├─────────────────────────────────────────────┤
  1. │ VW Diesel          41    2,040            0 │
  2. │ Subaru             35    2,050            0 │
  3. │ Datsun 210         35    2,020            0 │
  4. │ Plym. Arrow        28    3,260   .04429567 │
  5. │ Cad. Seville       21    4,290   .08241943 │
  6. │ Toyota Corolla     31    2,200   .10443129 │
  7. │ Olds 98            21    4,060   .28141296 │
     └─────────────────────────────────────────────┘
```

Being familiar with the automobile data, we immediately spotted two things: the VW is the only diesel car in our data, and the weight recorded for the Plymouth Arrow is incorrect.

❏

▷ Example 2

If we specify no explanatory variables, `rreg` produces a robust estimate of the mean:

```
. rreg mpg
   Huber iteration 1:  maximum difference in weights = .64471879
   Huber iteration 2:  maximum difference in weights = .05098336
   Huber iteration 3:  maximum difference in weights = .0099887
Biweight iteration 4:  maximum difference in weights = .25197391
Biweight iteration 5:  maximum difference in weights = .00358606

Robust regression                                     Number of obs =      74
                                                      F(  0,     73) =    0.00
                                                      Prob > F      =       .

──────────────────────────────────────────────────────────────────────────────
         mpg │      Coef.   Std. Err.      t    P>|t|     [95% Conf. Interval]
─────────────┼────────────────────────────────────────────────────────────────
       _cons │   20.68825    .641813    32.23   0.000     19.40912    21.96738
──────────────────────────────────────────────────────────────────────────────
```

The estimate is given by the coefficient on `_cons`. The mean is 20.69 with an estimated standard error of 0.6418. The 95% confidence interval is $[\,19.4, 22.0\,]$. By comparison, `ci` (see [R] **ci**) gives us the standard calculation:

```
. ci mpg

    Variable |        Obs        Mean    Std. Err.       [95% Conf. Interval]
-------------+---------------------------------------------------------------
         mpg |         74     21.2973    .6725511        19.9569     22.63769
```

◁

Stored results

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

Scalars	
`e(N)`	number of observations
`e(mss)`	model sum of squares
`e(df_m)`	model degrees of freedom
`e(rss)`	residual sum of squares
`e(df_r)`	residual degrees of freedom
`e(r2)`	R-squared
`e(r2_a)`	adjusted R-squared
`e(F)`	F statistic
`e(rmse)`	root mean squared error
`e(rank)`	rank of `e(V)`
Macros	
`e(cmd)`	`rreg`
`e(cmdline)`	command as typed
`e(depvar)`	name of dependent variable
`e(genwt)`	variable containing the weights
`e(title)`	title in estimation output
`e(model)`	`ols`
`e(vce)`	`ols`
`e(properties)`	`b V`
`e(predict)`	program used to implement `predict`
`e(marginsok)`	predictions allowed by `margins`
`e(asbalanced)`	factor variables `fvset` as `asbalanced`
`e(asobserved)`	factor variables `fvset` as `asobserved`
Matrices	
`e(b)`	coefficient vector
`e(V)`	variance–covariance matrix of the estimators
Functions	
`e(sample)`	marks estimation sample

Methods and formulas

See Berk (1990), Goodall (1983), and Rousseeuw and Leroy (1987) for a general description of the issues and methods. Hamilton (1991a, 1992) provides a more detailed description of `rreg` and some Monte Carlo evaluations.

`rreg` begins by fitting the regression (see [R] **regress**), calculating Cook's D (see [R] **predict** and [R] **regress postestimation**), and excluding any observation for which $D > 1$.

Thereafter `rreg` works iteratively: it performs a regression, calculates case weights from absolute residuals, and regresses again using those weights. Iterations stop when the maximum change in weights drops below `tolerance()`. Weights derive from one of two weight functions, Huber weights and biweights. Huber weights (Huber 1964) are used until convergence, and then, from that result, biweights are used until convergence. The biweight was proposed by Beaton and Tukey (1974, 151–152) after the Princeton robustness study (Andrews et al. 1972) had compared various estimators. Both weighting functions are used because Huber weights have problems dealing with severe outliers, whereas biweights sometimes fail to converge or have multiple solutions. The initial Huber weighting should improve the behavior of the biweight estimator.

In Huber weighting, cases with small residuals receive weights of 1; cases with larger residuals receive gradually smaller weights. Let $e_i = y_i - \mathbf{X}_i\mathbf{b}$ represent the ith-case residual. The ith scaled residual $u_i = e_i/s$ is calculated, where $s = M/0.6745$ is the residual scale estimate and $M = \text{med}(|e_i - \text{med}(e_i)|)$ is the median absolute deviation from the median residual. Huber estimation obtains case weights:

$$w_i = \begin{cases} 1 & \text{if } |u_i| \le c_h \\ c_h/|u_i| & \text{otherwise} \end{cases}$$

`rreg` defines $c_h = 1.345$, so downweighting begins with cases whose absolute residual exceeds $(1.345/0.6745)M \approx 2M$.

With biweights, all cases with nonzero residuals receive some downweighting, according to the smoothly decreasing biweight function

$$w_i = \begin{cases} \{1 - (u_i/c_b)^2\}^2 & \text{if } |u_i| \le c_b \\ 0 & \text{otherwise} \end{cases}$$

where $c_b = 4.685 \times$ `tune()`$/7$. Thus when `tune()` $= 7$, cases with absolute residuals of $(4.685/0.6745)M \approx 7M$ or more are assigned 0 weight and thus are effectively dropped. Goodall (1983, 377) suggests using a value between 6 and 9, inclusive, for `tune()` in the biweight case and states that performance is good between 6 and 12, inclusive.

The tuning constants $c_h = 1.345$ and $c_b = 4.685$ (assuming `tune()` is set at the default 7) give `rreg` about 95% of the efficiency of OLS when applied to data with normally distributed errors (Hamilton 1991b). Lower tuning constants downweight outliers more drastically (but give up Gaussian efficiency); higher tuning constants make the estimator more like OLS.

Standard errors are calculated using the pseudovalues approach described in Street, Carroll, and Ruppert (1988).

Acknowledgment

The current version of `rreg` is due to the work of Lawrence Hamilton of the Department of Sociology at the University of New Hampshire.

References

Andrews, D. F., P. J. Bickel, F. R. Hampel, P. J. Huber, W. H. Rogers, and J. W. Tukey. 1972. *Robust Estimates of Location: Survey and Advances*. Princeton: Princeton University Press.

Beaton, A. E., and J. W. Tukey. 1974. The fitting of power series, meaning polynomials, illustrated on band-spectroscopic data. *Technometrics* 16: 147–185.

Berk, R. A. 1990. A primer on robust regression. In *Modern Methods of Data Analysis*, ed. J. Fox and J. S. Long, 292–324. Newbury Park, CA: Sage.

Goodall, C. 1983. M-estimators of location: An outline of the theory. In *Understanding Robust and Exploratory Data Analysis*, ed. D. C. Hoaglin, C. F. Mosteller, and J. W. Tukey, 339–431. New York: Wiley.

Gould, W. W., and W. H. Rogers. 1994. Quantile regression as an alternative to robust regression. In *1994 Proceedings of the Statistical Computing Section*. Alexandria, VA: American Statistical Association.

Hamilton, L. C. 1991a. srd1: How robust is robust regression? *Stata Technical Bulletin* 2: 21–26. Reprinted in *Stata Technical Bulletin Reprints*, vol. 1, pp. 169–175. College Station, TX: Stata Press.

——. 1991b. ssi2: Bootstrap programming. *Stata Technical Bulletin* 4: 18–27. Reprinted in *Stata Technical Bulletin Reprints*, vol. 1, pp. 208–220. College Station, TX: Stata Press.

——. 1992. *Regression with Graphics: A Second Course in Applied Statistics*. Belmont, CA: Duxbury.

——. 2013. *Statistics with Stata: Updated for Version 12*. 8th ed. Boston: Brooks/Cole.

Huber, P. J. 1964. Robust estimation of a location parameter. *Annals of Mathematical Statistics* 35: 73–101.

Li, G. 1985. Robust regression. In *Exploring Data Tables, Trends, and Shapes*, ed. D. C. Hoaglin, C. F. Mosteller, and J. W. Tukey, 281–340. New York: Wiley.

Mosteller, C. F., and J. W. Tukey. 1977. *Data Analysis and Regression: A Second Course in Statistics*. Reading, MA: Addison–Wesley.

Relles, D. A., and W. H. Rogers. 1977. Statisticians are fairly robust estimators of location. *Journal of the American Statistical Association* 72: 107–111.

Rousseeuw, P. J., and A. M. Leroy. 1987. *Robust Regression and Outlier Detection*. New York: Wiley.

Street, J. O., R. J. Carroll, and D. Ruppert. 1988. A note on computing robust regression estimates via iteratively reweighted least squares. *American Statistician* 42: 152–154.

Verardi, V., and C. Croux. 2009. Robust regression in Stata. *Stata Journal* 9: 439–453.

Also see

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

[R] **qreg** — Quantile regression

[R] **regress** — Linear regression

[MI] **estimation** — Estimation commands for use with mi estimate

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