Title

regress — Linear regression

Syntax

`regress` *depvar* [*indepvars*] [*if*] [*in*] [*weight*] [, *options*]

options	Description
Model	
`noconstant`	suppress constant term
`hascons`	has user-supplied constant
`tsscons`	compute total sum of squares with constant; seldom used
SE/Robust	
`vce(`*vcetype*`)`	*vcetype* may be `ols`, `robust`, `cluster` *clustvar*, `bootstrap`, `jackknife`, `hc2`, or `hc3`
Reporting	
`level(`*#*`)`	set confidence level; default is `level(95)`
`beta`	report standardized beta coefficients
`eform(`*string*`)`	report exponentiated coefficients and label as *string*
`depname(`*varname*`)`	substitute dependent variable name; programmer's option
display_options	control column formats, row spacing, line width, display of omitted variables and base and empty cells, and factor-variable labeling
`noheader`	suppress output header
`notable`	suppress coefficient table
`plus`	make table extendable
`mse1`	force mean squared error to 1
`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**.

`bootstrap`, `by`, `fp`, `jackknife`, `mfp`, `mi estimate`, `nestreg`, `rolling`, `statsby`, `stepwise`, and `svy` are allowed; see **[U] 11.1.10 Prefix commands**.

`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**.

`aweight`s are not allowed with the `jackknife` prefix; see **[R] jackknife**.

`hascons`, `tsscons`, `vce()`, `beta`, `noheader`, `notable`, `plus`, `depname()`, `mse1`, and weights are not allowed with the `svy` prefix; see **[SVY] svy**.

`aweight`s, `fweight`s, `iweight`s, and `pweight`s are allowed; see **[U] 11.1.6 weight**.

`noheader`, `notable`, `plus`, `mse1`, and `coeflegend` do 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 > Linear regression

Description

`regress` fits a model of *depvar* on *indepvars* using linear regression.

Here is a short list of other regression commands that may be of interest. See `help estimation commands` for a complete list.

Command	Entry	Description
`areg`	[R] **areg**	an easier way to fit regressions with many dummy variables
`arch`	[TS] **arch**	regression models with ARCH errors
`arima`	[TS] **arima**	ARIMA models
`boxcox`	[R] **boxcox**	Box–Cox regression models
`cnsreg`	[R] **cnsreg**	constrained linear regression
`eivreg`	[R] **eivreg**	errors-in-variables regression
`etregress`	[TE] **etregress**	Linear regression with endogenous treatment effects
`frontier`	[R] **frontier**	stochastic frontier models
`gmm`	[R] **gmm**	generalized method of moments estimation
`heckman`	[R] **heckman**	Heckman selection model
`intreg`	[R] **intreg**	interval regression
`ivregress`	[R] **ivregress**	single-equation instrumental-variables regression
`ivtobit`	[R] **ivtobit**	tobit regression with endogenous variables
`newey`	[TS] **newey**	regression with Newey–West standard errors
`nl`	[R] **nl**	nonlinear least-squares estimation
`nlsur`	[R] **nlsur**	estimation of nonlinear systems of equations
`qreg`	[R] **qreg**	quantile (including median) regression
`reg3`	[R] **reg3**	three-stage least-squares (3SLS) regression
`rreg`	[R] **rreg**	a type of robust regression
`gsem`	[SEM] **intro 5**	generalized structural equation models
`sem`	[SEM] **intro 5**	linear structural equation models
`sureg`	[R] **sureg**	seemingly unrelated regression
`tobit`	[R] **tobit**	tobit regression
`truncreg`	[R] **truncreg**	truncated regression
`xtabond`	[XT] **xtabond**	Arellano–Bond linear dynamic panel-data estimation
`xtdpd`	[XT] **xtdpd**	linear dynamic panel-data estimation
`xtfrontier`	[XT] **xtfrontier**	panel-data stochastic frontier models
`xtgls`	[XT] **xtgls**	panel-data GLS models
`xthtaylor`	[XT] **xthtaylor**	Hausman–Taylor estimator for error-components models
`xtintreg`	[XT] **xtintreg**	panel-data interval regression models
`xtivreg`	[XT] **xtivreg**	panel-data instrumental-variables (2SLS) regression
`xtpcse`	[XT] **xtpcse**	linear regression with panel-corrected standard errors
`xtreg`	[XT] **xtreg**	fixed- and random-effects linear models
`xtregar`	[XT] **xtregar**	fixed- and random-effects linear models with an AR(1) disturbance
`xttobit`	[XT] **xttobit**	panel-data tobit models

Options

Model

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

`hascons` indicates that a user-defined constant or its equivalent is specified among the independent variables in *indepvars*. Some caution is recommended when specifying this option, as resulting estimates may not be as accurate as they otherwise would be. Use of this option requires "sweeping" the constant last, so the moment matrix must be accumulated in absolute rather than deviation form. This option may be safely specified when the means of the dependent and independent variables are all reasonable and there is not much collinearity between the independent variables. The best procedure is to view `hascons` as a reporting option—estimate with and without `hascons` and verify that the coefficients and standard errors of the variables not affected by the identity of the constant are unchanged.

`tsscons` forces the total sum of squares to be computed as though the model has a constant, that is, as deviations from the mean of the dependent variable. This is a rarely used option that has an effect only when specified with `noconstant`. It affects the total sum of squares and all results derived from the total sum of squares.

SE/Robust

`vce(`*vcetype*`)` specifies the type of standard error reported, which includes types that are derived from asymptotic theory (`ols`), 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***.

`vce(ols)`, the default, uses the standard variance estimator for ordinary least-squares regression.

`regress` also allows the following:

`vce(hc2)` and `vce(hc3)` specify an alternative bias correction for the robust variance calculation. `vce(hc2)` and `vce(hc3)` may not be specified with `svy` prefix. In the unclustered case, `vce(robust)` uses $\widehat{\sigma}_j^2 = \{n/(n-k)\}u_j^2$ as an estimate of the variance of the jth observation, where u_j is the calculated residual and $n/(n-k)$ is included to improve the overall estimate's small-sample properties.

`vce(hc2)` instead uses $u_j^2/(1-h_{jj})$ as the observation's variance estimate, where h_{jj} is the diagonal element of the hat (projection) matrix. This estimate is unbiased if the model really is homoskedastic. `vce(hc2)` tends to produce slightly more conservative confidence intervals.

`vce(hc3)` uses $u_j^2/(1-h_{jj})^2$ as suggested by Davidson and MacKinnon (1993), who report that this method tends to produce better results when the model really is heteroskedastic. `vce(hc3)` produces confidence intervals that tend to be even more conservative.

See Davidson and MacKinnon (1993, 554–556) and Angrist and Pischke (2009, 294–308) for more discussion on these two bias corrections.

Reporting

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

`beta` asks that standardized beta coefficients be reported instead of confidence intervals. The beta coefficients are the regression coefficients obtained by first standardizing all variables to have a mean of 0 and a standard deviation of 1. `beta` may not be specified with `vce(cluster` *clustvar*`)` or the `svy` prefix.

`eform(`*string*`)` is used only in programs and ado-files that use `regress` to fit models other than linear regression. `eform()` specifies that the coefficient table be displayed in exponentiated form as defined in [R] **maximize** and that *string* be used to label the exponentiated coefficients in the table.

`depname(`*varname*`)` is used only in programs and ado-files that use `regress` to fit models other than linear regression. `depname()` may be specified only at estimation time. *varname* is recorded as the identity of the dependent variable, even though the estimates are calculated using *depvar*. This method affects the labeling of the output—not the results calculated—but could affect subsequent calculations made by `predict`, where the residual would be calculated as deviations from *varname* rather than *depvar*. `depname()` is most typically used when *depvar* is a temporary variable (see [P] **macro**) used as a proxy for *varname*.

`depname()` is not allowed with the `svy` prefix.

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

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

`noheader` suppresses the display of the ANOVA table and summary statistics at the top of the output; only the coefficient table is displayed. This option is often used in programs and ado-files.

`notable` suppresses display of the coefficient table.

`plus` specifies that the output table be made extendable. This option is often used in programs and ado-files.

`mse1` is used only in programs and ado-files that use `regress` to fit models other than linear regression and is not allowed with the `svy` prefix. `mse1` sets the mean squared error to 1, forcing the variance–covariance matrix of the estimators to be $(\mathbf{X}'\mathbf{D}\mathbf{X})^{-1}$ (see *Methods and formulas* below) and affecting calculated standard errors. Degrees of freedom for t statistics is calculated as n rather than $n-k$.

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

Remarks and examples

Remarks are presented under the following headings:

`regress` performs linear regression, including ordinary least squares and weighted least squares. For a general discussion of linear regression, see Draper and Smith (1998), Greene (2012), or Kmenta (1997).

See Wooldridge (2013) for an excellent treatment of estimation, inference, interpretation, and specification testing in linear regression models. This presentation stands out for its clarification of the statistical issues, as opposed to the algebraic issues. See Wooldridge (2010, chap. 4) for a more advanced discussion along the same lines.

See Hamilton (2013, chap. 7) and Cameron and Trivedi (2010, chap. 3) for an introduction to linear regression using Stata. Dohoo, Martin, and Stryhn (2012, 2010) discuss linear regression using examples from epidemiology, and Stata datasets and do-files used in the text are available. Cameron and Trivedi (2010) discuss linear regression using econometric examples with Stata. Mitchell (2012) shows how to use graphics and postestimation commands to understand a fitted regression model.

Chatterjee and Hadi (2012) explain regression analysis by using examples containing typical problems that you might encounter when performing exploratory data analysis. We also recommend Weisberg (2005), who emphasizes the importance of the assumptions of linear regression and problems resulting from these assumptions. Becketti (2013) discusses regression analysis with an emphasis on time-series data. Angrist and Pischke (2009) approach regression as a tool for exploring relationships, estimating treatment effects, and providing answers to public policy questions. For a discussion of model-selection techniques and exploratory data analysis, see Mosteller and Tukey (1977). For a mathematically rigorous treatment, see Peracchi (2001, chap. 6). Finally, see Plackett (1972) if you are interested in the history of regression. Least squares, which dates back to the 1790s, was discovered independently by Legendre and Gauss.

Ordinary least squares

▷ Example 1: Basic linear regression

Suppose that we have data on the mileage rating and weight of 74 automobiles. The variables in our data are `mpg`, `weight`, and `foreign`. The last variable assumes the value 1 for foreign and 0 for domestic automobiles. We wish to fit the model

$$\texttt{mpg} = \beta_0 + \beta_1\texttt{weight} + \beta_2\texttt{foreign} + \epsilon$$

This model can be fit with `regress` by typing

```
. 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
------------------------------------------------------------------------------
```

`regress` produces a variety of summary statistics along with the table of regression coefficients. At the upper left, `regress` reports an analysis-of-variance (ANOVA) table. The column headings `SS`, `df`, and `MS` stand for "sum of squares", "degrees of freedom", and "mean square", respectively. In this example, the total sum of squares is 2,443.5: 1,619.3 accounted for by the model and 824.2 left unexplained. Because the regression included a constant, the total sum reflects the sum after removal of means, as does the sum of squares due to the model. The table also reveals that there are 73 total degrees of freedom (counted as 74 observations less 1 for the mean removal), of which 2 are consumed by the model, leaving 71 for the residual.

To the right of the ANOVA table are presented other summary statistics. The F statistic associated with the ANOVA table is 69.75. The statistic has 2 numerator and 71 denominator degrees of freedom. The F statistic tests the hypothesis that all coefficients excluding the constant are zero. The chance of observing an F statistic that large or larger is reported as 0.0000, which is Stata's way of indicating a number smaller than 0.00005. The R-squared (R^2) for the regression is 0.6627, and the R-squared adjusted for degrees of freedom (R_a^2) is 0.6532. The root mean squared error, labeled `Root MSE`, is 3.4071. It is the square root of the mean squared error reported for the residual in the ANOVA table.

Finally, Stata produces a table of the estimated coefficients. The first line of the table indicates that the left-hand-side variable is `mpg`. Thereafter follow the estimated coefficients. Our fitted model is

$$\texttt{mpg\_hat} = 41.68 - 0.0066\,\texttt{weight} - 1.65\,\texttt{foreign}$$

Reported to the right of the coefficients in the output are the standard errors. For instance, the standard error for the coefficient on `weight` is 0.0006371. The corresponding t statistic is -10.34, which has a two-sided significance level of 0.000. This number indicates that the significance is less than 0.0005. The 95% confidence interval for the coefficient is $[-0.0079, -0.0053]$.

◁

▷ Example 2: Transforming the dependent variable

If we had a graph comparing `mpg` with `weight`, we would notice that the relationship is distinctly nonlinear. This is to be expected because energy usage per distance should increase linearly with weight, but `mpg` is measuring distance per energy used. We could obtain a better model by generating a new variable measuring the number of gallons used per 100 miles (`gp100m`) and then using this new variable in our model:

$$\texttt{gp100m} = \beta_0 + \beta_1\texttt{weight} + \beta_2\texttt{foreign} + \epsilon$$

We can now fit this model:

```
. generate gp100m = 100/mpg
. regress gp100m weight foreign
      Source |       SS           df       MS      Number of obs   =        74
-------------+----------------------------------   F(2, 71)        =    113.97
       Model |  91.1761694         2  45.5880847   Prob > F        =    0.0000
    Residual |  28.4000913        71  .400001287   R-squared       =    0.7625
-------------+----------------------------------   Adj R-squared   =    0.7558
       Total |  119.576261        73  1.63803097   Root MSE        =    .63246

------------------------------------------------------------------------------
      gp100m |      Coef.   Std. Err.      t    P>|t|     [95% Conf. Interval]
-------------+----------------------------------------------------------------
      weight |   .0016254   .0001183    13.74   0.000     .0013896    .0018612
     foreign |   .6220535   .1997381     3.11   0.003     .2237871     1.02032
       _cons |  -.0734839   .4019932    -0.18   0.855    -.8750354    .7280677
------------------------------------------------------------------------------
```

Fitting the physically reasonable model increases our R-squared to 0.7625.

◁

▷ Example 3: Obtaining beta coefficients

`regress` shares the features of all estimation commands. Among other things, this means that after running a regression, we can use `test` to test hypotheses about the coefficients, `estat vce` to examine the covariance matrix of the estimators, and `predict` to obtain predicted values, residuals, and influence statistics. See [U] 20 Estimation and postestimation commands. Options that affect how estimates are displayed, such as `beta` or `level()`, can be used when replaying results.

Suppose that we meant to specify the `beta` option to obtain beta coefficients (regression coefficients normalized by the ratio of the standard deviation of the regressor to the standard deviation of the dependent variable). Even though we forgot, we can specify the option now:

```
. regress, beta

      Source |       SS           df       MS      Number of obs   =        74
-------------+----------------------------------   F(2, 71)        =    113.97
       Model |  91.1761694         2  45.5880847   Prob > F        =    0.0000
    Residual |  28.4000913        71  .400001287   R-squared       =    0.7625
-------------+----------------------------------   Adj R-squared   =    0.7558
       Total |  119.576261        73  1.63803097   Root MSE        =    .63246

------------------------------------------------------------------------------
      gp100m |      Coef.   Std. Err.      t    P>|t|                     Beta
-------------+----------------------------------------------------------------
      weight |   .0016254   .0001183    13.74   0.000                 .9870255
     foreign |   .6220535   .1997381     3.11   0.003                 .2236673
       _cons |  -.0734839   .4019932    -0.18   0.855                        .
------------------------------------------------------------------------------
```

◁

Treatment of the constant

By default, `regress` includes an intercept (constant) term in the model. The `noconstant` option suppresses it, and the `hascons` option tells `regress` that the model already has one.

▷ Example 4: Suppressing the constant term

We wish to fit a regression of the `weight` of an automobile against its `length`, and we wish to impose the constraint that the weight is zero when the length is zero.

If we simply type `regress weight length`, we are fitting the model

$$\texttt{weight} = \beta_0 + \beta_1\,\texttt{length} + \epsilon$$

Here a `length` of zero corresponds to a `weight` of β_0. We want to force β_0 to be zero or, equivalently, estimate an equation that does not include an intercept:

$$\texttt{weight} = \beta_1\,\texttt{length} + \epsilon$$

We do this by specifying the `noconstant` option:

```
. regress weight length, noconstant

      Source |       SS           df       MS      Number of obs   =        74
-------------+----------------------------------   F(1, 73)        =   3450.13
       Model |   703869302         1   703869302   Prob > F        =    0.0000
    Residual |  14892897.8        73  204012.299   R-squared       =    0.9793
-------------+----------------------------------   Adj R-squared   =    0.9790
       Total |   718762200        74   9713002.7   Root MSE        =    451.68

------------------------------------------------------------------------------
      weight |      Coef.   Std. Err.      t    P>|t|     [95% Conf. Interval]
-------------+----------------------------------------------------------------
      length |   16.29829   .2774752    58.74   0.000     15.74528     16.8513
------------------------------------------------------------------------------
```

In our data, `length` is measured in inches and `weight` in pounds. We discover that each inch of length adds 16 pounds to the weight.

◁

Sometimes there is no need for Stata to include a constant term in the model. Most commonly, this occurs when the model contains a set of mutually exclusive indicator variables. `hascons` is a variation of the `noconstant` option—it tells Stata not to add a constant to the regression because the regression specification already has one, either directly or indirectly.

For instance, we now refit our model of `weight` as a function of `length` and include separate constants for foreign and domestic cars by specifying `bn.foreign`. `bn.foreign` is factor-variable notation for “no base for `foreign`” or “include all levels of variable `foreign` in the model”; see [U] **11.4.3 Factor variables**.

```
. regress weight length bn.foreign, hascons

      Source |       SS           df       MS      Number of obs   =        74
-------------+----------------------------------   F(2, 71)        =    316.54
       Model |  39647744.7         2  19823872.3   Prob > F        =    0.0000
    Residual |   4446433.7        71  62625.8268   R-squared       =    0.8992
-------------+----------------------------------   Adj R-squared   =    0.8963
       Total |  44094178.4        73  604029.841   Root MSE        =    250.25

------------------------------------------------------------------------------
      weight |      Coef.   Std. Err.      t    P>|t|     [95% Conf. Interval]
-------------+----------------------------------------------------------------
      length |   31.44455   1.601234    19.64   0.000     28.25178    34.63732
             |
     foreign |
   Domestic  |   -2850.25   315.9691    -9.02   0.000    -3480.274   -2220.225
    Foreign  |  -2983.927   275.1041   -10.85   0.000    -3532.469   -2435.385
------------------------------------------------------------------------------
```

❑ Technical note

There is a subtle distinction between the `hascons` and `noconstant` options. We can most easily reveal it by refitting the last regression, specifying `noconstant` rather than `hascons`:

```
. regress weight length bn.foreign, noconstant

      Source |       SS           df       MS      Number of obs   =        74
-------------+----------------------------------   F(  3,     71)  =   3802.03
       Model |   714315766         3   238105255   Prob > F        =    0.0000
    Residual |   4446433.7        71  62625.8268   R-squared       =    0.9938
-------------+----------------------------------   Adj R-squared   =    0.9936
       Total |   718762200        74   9713002.7   Root MSE        =    250.25

------------------------------------------------------------------------------
      weight |      Coef.   Std. Err.      t    P>|t|     [95% Conf. Interval]
-------------+----------------------------------------------------------------
      length |   31.44455   1.601234    19.64   0.000     28.25178    34.63732
             |
     foreign |
   Domestic  |   -2850.25   315.9691    -9.02   0.000    -3480.274   -2220.225
   Foreign   |  -2983.927   275.1041   -10.85   0.000    -3532.469   -2435.385
------------------------------------------------------------------------------
```

Comparing this output with that produced by the previous `regress` command, we see that they are almost, but not quite, identical. The parameter estimates and their associated statistics—the second half of the output—are identical. The overall summary statistics and the ANOVA table—the first half of the output—are different, however.

In the first case, the R^2 is shown as 0.8992; here it is shown as 0.9938. In the first case, the F statistic is 316.54; now it is 3,802.03. The numerator degrees of freedom is different as well. In the first case, the numerator degrees of freedom is 2; now the degrees of freedom is 3. Which is correct?

Both are. Specifying the `hascons` option causes `regress` to adjust the ANOVA table and its associated statistics for the explanatory power of the constant. The regression in effect has a constant; it is just written in such a way that a separate constant is unnecessary. No such adjustment is made with the `noconstant` option.

❑

❑ Technical note

When the `hascons` option is specified, `regress` checks to make sure that the model does in fact have a constant term. If `regress` cannot find a constant term, it automatically adds one. Fitting a model of `weight` on `length` and specifying the `hascons` option, we obtain

```
. regress weight length, hascons
(note: hascons false)

      Source |       SS           df       MS      Number of obs   =        74
-------------+----------------------------------   F(  1,     72)  =    613.27
       Model |  39461306.8         1  39461306.8   Prob > F        =    0.0000
    Residual |  4632871.55        72  64345.4382   R-squared       =    0.8949
-------------+----------------------------------   Adj R-squared   =    0.8935
       Total |  44094178.4        73  604029.841   Root MSE        =    253.66

------------------------------------------------------------------------------
      weight |      Coef.   Std. Err.      t    P>|t|     [95% Conf. Interval]
-------------+----------------------------------------------------------------
      length |   33.01988   1.333364    24.76   0.000     30.36187    35.67789
       _cons |  -3186.047   252.3113   -12.63   0.000     -3689.02   -2683.073
------------------------------------------------------------------------------
```

Even though we specified `hascons`, `regress` included a constant, anyway. It also added a note to our output: “note: hascons false”.

❑

❑ Technical note

Even if the model specification effectively includes a constant term, we need not specify the `hascons` option. `regress` is always on the lookout for collinear variables and omits them from the model. For instance,

```
. regress weight length bn.foreign
note: 1.foreign omitted because of collinearity
      Source |       SS           df       MS      Number of obs   =        74
-------------+----------------------------------   F(  2,     71)   =    316.54
       Model |  39647744.7         2  19823872.3   Prob > F        =    0.0000
    Residual |   4446433.7        71  62625.8268   R-squared       =    0.8992
-------------+----------------------------------   Adj R-squared   =    0.8963
       Total |  44094178.4        73  604029.841   Root MSE        =    250.25

------------------------------------------------------------------------------
      weight |      Coef.   Std. Err.      t    P>|t|     [95% Conf. Interval]
-------------+----------------------------------------------------------------
      length |   31.44455   1.601234    19.64   0.000     28.25178    34.63732
             |
     foreign |
   Domestic  |   133.6775   77.47615     1.73   0.089    -20.80555    288.1605
    Foreign  |          0  (omitted)
             |
       _cons |  -2983.927   275.1041   -10.85   0.000    -3532.469   -2435.385
------------------------------------------------------------------------------
```

❑

Robust standard errors

`regress` with the `vce(robust)` option substitutes a robust variance matrix calculation for the conventional calculation, or if `vce(cluster` *clustvar*`)` is specified, allows relaxing the assumption of independence within groups. How this method works is explained in **[U] 20.21 Obtaining robust variance estimates**. Below we show how well this approach works.

▷ Example 5: Heteroskedasticity and robust standard errors

Specifying the `vce(robust)` option is equivalent to requesting White-corrected standard errors in the presence of heteroskedasticity. We use the automobile data and, in the process of looking at the energy efficiency of cars, analyze a variable with considerable heteroskedasticity.

We will examine the amount of energy—measured in gallons of gasoline—that the cars in the data need to move 1,000 pounds of their weight 100 miles. We are going to examine the relative efficiency of foreign and domestic cars.

```
. gen gpmw = ((1/mpg)/weight)*100*1000
. summarize gpmw
    Variable |        Obs        Mean    Std. Dev.       Min        Max
-------------+---------------------------------------------------------
        gpmw |         74    1.682184    .2426311    1.09553    2.30521
```

In these data, the engines consume between 1.10 and 2.31 gallons of gas to move 1,000 pounds of the car’s weight 100 miles. If we ran a regression with conventional standard errors of `gpmw` on `foreign`, we would obtain

```
. regress gpmw foreign

      Source |       SS       df       MS              Number of obs =      74
-------------+------------------------------           F(  1,    72) =   20.07
       Model |  .936705572     1  .936705572           Prob > F      =  0.0000
    Residual |  3.36079459    72  .046677703           R-squared     =  0.2180
-------------+------------------------------           Adj R-squared =  0.2071
       Total |  4.29750017    73  .058869865           Root MSE      =  .21605

------------------------------------------------------------------------------
        gpmw |      Coef.   Std. Err.      t    P>|t|     [95% Conf. Interval]
-------------+----------------------------------------------------------------
     foreign |   .2461526   .0549487     4.48   0.000     .1366143    .3556909
       _cons |   1.609004   .0299608    53.70   0.000     1.549278     1.66873
------------------------------------------------------------------------------
```

`regress` with the `vce(robust)` option, on the other hand, reports

```
. regress gpmw foreign, vce(robust)

Linear regression                                      Number of obs =      74
                                                       F(  1,    72) =   13.13
                                                       Prob > F      =  0.0005
                                                       R-squared     =  0.2180
                                                       Root MSE      =  .21605

------------------------------------------------------------------------------
             |               Robust
        gpmw |      Coef.   Std. Err.      t    P>|t|     [95% Conf. Interval]
-------------+----------------------------------------------------------------
     foreign |   .2461526   .0679238     3.62   0.001     .1107489    .3815563
       _cons |   1.609004   .0234535    68.60   0.000      1.56225    1.655758
------------------------------------------------------------------------------
```

The point estimates are the same (foreign cars need one-quarter gallon more gas), but the standard errors differ by roughly 20%. Conventional regression reports the 95% confidence interval as $[0.14, 0.36]$, whereas the robust standard errors make the interval $[0.11, 0.38]$.

Which is right? Notice that `gpmw` is a variable with considerable heteroskedasticity:

```
. tabulate foreign, summarize(gpmw)

            |           Summary of gpmw
   Car type |        Mean   Std. Dev.       Freq.
------------+------------------------------------
   Domestic |   1.6090039   .16845182          52
    Foreign |   1.8551565   .30186861          22
------------+------------------------------------
      Total |   1.6821844   .24263113          74
```

Thus here we favor the robust standard errors. In **[U] 20.21 Obtaining robust variance estimates**, we show another example using linear regression where it makes little difference whether we specify `vce(robust)`. The linear-regression assumptions were true, and we obtained nearly linear-regression results. The advantage of the robust estimate is that in neither case did we have to check assumptions.

◁

❑ Technical note

`regress` purposefully suppresses displaying the ANOVA table when `vce(robust)` is specified, as it is no longer appropriate in a statistical sense, even though, mechanically, the numbers would be unchanged. That is, sums of squares remain unchanged, but the meaning of those sums is no longer relevant. The F statistic, for instance, is no longer based on sums of squares; it becomes a Wald test based on the robustly estimated variance matrix. Nevertheless, `regress` continues to report the R^2

and the root MSE even though both numbers are based on sums of squares and are, strictly speaking, irrelevant. In this, the root MSE is more in violation of the spirit of the robust estimator than is R^2. As a goodness-of-fit statistic, R^2 is still fine; just do not use it in formulas to obtain F statistics because those formulas no longer apply. The root MSE is valid in a literal sense—it is the square root of the mean squared error, but it is no longer an estimate of σ because there is no single σ; the variance of the residual varies observation by observation.

❑

▷ Example 6: Alternative robust standard errors

The `vce(hc2)` and `vce(hc3)` options modify the robust variance calculation. In the context of linear regression without clustering, the idea behind the robust calculation is somehow to measure σ_j^2, the variance of the residual associated with the jth observation, and then to use that estimate to improve the estimated variance of $\widehat{\boldsymbol{\beta}}$. Because residuals have (theoretically and practically) mean 0, one estimate of σ_j^2 is the observation's squared residual itself—u_j^2. A finite-sample correction could improve that by multiplying u_j^2 by $n/(n-k)$, and, as a matter of fact, `vce(robust)` uses $\{n/(n-k)\}u_j^2$ as its estimate of the residual's variance.

`vce(hc2)` and `vce(hc3)` use alternative estimators of the observation-specific variances. For instance, if the residuals are homoskedastic, we can show that the expected value of u_j^2 is $\sigma^2(1-h_{jj})$, where h_{jj} is the jth diagonal element of the projection (hat) matrix. h_{jj} has average value k/n, so $1-h_{jj}$ has average value $1-k/n = (n-k)/n$. Thus the default robust estimator $\widehat{\sigma}_j = \{n/(n-k)\}u_j^2$ amounts to dividing u_j^2 by the average of the expectation.

`vce(hc2)` divides u_j^2 by $1-h_{jj}$ itself, so it should yield better estimates if the residuals really are homoskedastic. `vce(hc3)` divides u_j^2 by $(1-h_{jj})^2$ and has no such clean interpretation. Davidson and MacKinnon (1993) show that $u_j^2/(1-h_{jj})^2$ approximates a more complicated estimator that they obtain by jackknifing (MacKinnon and White 1985). Angrist and Pischke (2009) also illustrate the relative merits of these adjustments.

Here are the results of refitting our efficiency model using `vce(hc2)` and `vce(hc3)`:

```
. regress gpmw foreign, vce(hc2)

Linear regression                                      Number of obs =      74
                                                       F(  1,     72) =   12.93
                                                       Prob > F      =  0.0006
                                                       R-squared     =  0.2180
                                                       Root MSE      =  .21605

------------------------------------------------------------------------------
             |             Robust HC2
        gpmw |      Coef.   Std. Err.      t    P>|t|     [95% Conf. Interval]
-------------+----------------------------------------------------------------
     foreign |   .2461526   .0684669     3.60   0.001     .1096662    .3826389
       _cons |   1.609004   .0233601    68.88   0.000     1.562437    1.655571
------------------------------------------------------------------------------
```

```
. regress gpmw foreign, vce(hc3)
Linear regression                                      Number of obs =      74
                                                       F(  1,     72) =   12.38
                                                       Prob > F      =  0.0008
                                                       R-squared     =  0.2180
                                                       Root MSE      =  .21605

------------------------------------------------------------------------------
             |             Robust HC3
        gpmw |      Coef.   Std. Err.      t    P>|t|     [95% Conf. Interval]
-------------+----------------------------------------------------------------
     foreign |   .2461526    .069969     3.52   0.001     .1066719    .3856332
       _cons |   1.609004    .023588    68.21   0.000     1.561982    1.656026
------------------------------------------------------------------------------
```

Example 7: Standard errors for clustered data

The `vce(cluster` *clustvar*`)` option relaxes the assumption of independence. Below we have 28,534 observations on 4,711 women aged 14–46 years. Data were collected on these women between 1968 and 1988. We are going to fit a classic earnings model, and we begin by ignoring that each woman appears an average of 6.057 times in the data.

```
. use http://www.stata-press.com/data/r13/regsmpl, clear
(NLS Women 14-26 in 1968)
. regress ln_wage age c.age#c.age tenure
      Source |       SS       df       MS              Number of obs =   28101
-------------+------------------------------           F(  3, 28097) = 1842.45
       Model |  1054.52501     3  351.508335           Prob > F      =  0.0000
    Residual |  5360.43962 28097  .190783344           R-squared     =  0.1644
-------------+------------------------------           Adj R-squared =  0.1643
       Total |  6414.96462 28100  .228290556           Root MSE      =  .43679

------------------------------------------------------------------------------
     ln_wage |      Coef.   Std. Err.      t    P>|t|     [95% Conf. Interval]
-------------+----------------------------------------------------------------
         age |   .0752172   .0034736    21.65   0.000     .0684088    .0820257
             |
 c.age#c.age |  -.0010851   .0000575   -18.86   0.000    -.0011979   -.0009724
             |
      tenure |   .0390877   .0007743    50.48   0.000     .0375699    .0406054
       _cons |   .3339821   .0504413     6.62   0.000     .2351148    .4328495
------------------------------------------------------------------------------
```

The number of observations in our model is 28,101 because Stata drops observations that have a missing value for one or more of the variables in the model. We can be reasonably certain that the standard errors reported above are meaningless. Without a doubt, a woman with higher-than-average wages in one year typically has higher-than-average wages in other years, and so the residuals are not independent. One way to deal with this would be to fit a random-effects model—and we are going to do that—but first we fit the model using `regress` specifying `vce(cluster id)`, which treats only observations with different person `id`s as truly independent:

```
. regress ln_wage age c.age#c.age tenure, vce(cluster id)
Linear regression                                      Number of obs =   28101
                                                       F(  3,  4698) =  748.82
                                                       Prob > F      =  0.0000
                                                       R-squared     =  0.1644
                                                       Root MSE      =  .43679
                                (Std. Err. adjusted for 4699 clusters in idcode)
------------------------------------------------------------------------------
             |               Robust
     ln_wage |      Coef.   Std. Err.      t    P>|t|     [95% Conf. Interval]
-------------+----------------------------------------------------------------
         age |   .0752172   .0045711    16.45   0.000     .0662557    .0841788
             |
 c.age#c.age |  -.0010851   .0000778   -13.94   0.000    -.0012377   -.0009325
             |
      tenure |   .0390877   .0014425    27.10   0.000     .0362596    .0419157
       _cons |   .3339821   .0641918     5.20   0.000      .208136    .4598282
------------------------------------------------------------------------------
```

For comparison, we focus on the tenure coefficient, which in economics jargon can be interpreted as the rate of return for keeping your job. The 95% confidence interval we previously estimated—an interval we do not believe—is $[\,0.038, 0.041\,]$. The robust interval is twice as wide, being $[\,0.036, 0.042\,]$.

As we said, one correct way to fit this model is by random-effects regression. Here is the random-effects result:

```
. xtreg ln_wage age c.age#c.age tenure, re
Random-effects GLS regression                   Number of obs      =     28101
Group variable: idcode                          Number of groups   =      4699
R-sq:  within  = 0.1370                         Obs per group: min =         1
       between = 0.2154                                        avg =       6.0
       overall = 0.1608                                        max =        15
                                                Wald chi2(3)       =   4717.05
corr(u_i, X)   = 0 (assumed)                    Prob > chi2        =    0.0000
------------------------------------------------------------------------------
     ln_wage |      Coef.   Std. Err.      z    P>|z|     [95% Conf. Interval]
-------------+----------------------------------------------------------------
         age |   .0568296   .0026958    21.08   0.000     .0515459    .0621132
             |
 c.age#c.age |  -.0007566   .0000447   -16.93   0.000    -.0008441    -.000669
             |
      tenure |   .0260135   .0007477    34.79   0.000     .0245481    .0274789
       _cons |   .6136792   .0394611    15.55   0.000     .5363368    .6910216
-------------+----------------------------------------------------------------
     sigma_u |  .33542449
     sigma_e |  .29674679
         rho |  .56095413   (fraction of variance due to u_i)
------------------------------------------------------------------------------
```

Robust regression estimated the 95% interval $[\,0.036, 0.042\,]$, and `xtreg` (see [XT] **xtreg**) estimates $[\,0.025, 0.027\,]$. Which is better? The random-effects regression estimator assumes a lot. We can check some of these assumptions by performing a Hausman test. Using `estimates` (see [R] **estimates store**), we store the random-effects estimation results, and then we run the required fixed-effects regression to perform the test.

```
. estimates store random

. xtreg ln_wage age c.age#c.age tenure, fe

Fixed-effects (within) regression               Number of obs      =     28101
Group variable: idcode                          Number of groups   =      4699

R-sq:  within  = 0.1375                         Obs per group: min =         1
       between = 0.2066                                        avg =       6.0
       overall = 0.1568                                        max =        15

                                                F(3,23399)         =   1243.00
corr(u_i, Xb)  = 0.1380                         Prob > F           =    0.0000

------------------------------------------------------------------------------
     ln_wage |      Coef.   Std. Err.      t    P>|t|     [95% Conf. Interval]
-------------+----------------------------------------------------------------
         age |   .0522751    .002783    18.78   0.000     .0468202      .05773
             |
 c.age#c.age |  -.0006717   .0000461   -14.56   0.000    -.0007621   -.0005813
             |
      tenure |    .021738    .000799    27.21   0.000      .020172     .023304
       _cons |    .687178   .0405944    16.93   0.000     .6076103    .7667456
-------------+----------------------------------------------------------------
     sigma_u |  .38743138
     sigma_e |  .29674679
         rho |   .6302569   (fraction of variance due to u_i)
------------------------------------------------------------------------------
F test that all u_i=0:     F(4698, 23399) =     7.98         Prob > F = 0.0000

. hausman . random

                 ---- Coefficients ----
             |      (b)          (B)            (b-B)     sqrt(diag(V_b-V_B))
             |       .          random       Difference          S.E.
-------------+----------------------------------------------------------------
         age |    .0522751     .0568296       -.0045545        .0006913
 c.age#c.age |   -.0006717    -.0007566        .0000849        .0000115
      tenure |     .021738     .0260135       -.0042756        .0002816
------------------------------------------------------------------------------
                           b = consistent under Ho and Ha; obtained from xtreg
            B = inconsistent under Ha, efficient under Ho; obtained from xtreg

    Test:  Ho:  difference in coefficients not systematic

                  chi2(3) = (b-B)'[(V_b-V_B)^(-1)](b-B)
                          =      336.62
                Prob>chi2 =      0.0000
```

The Hausman test casts grave suspicions on the random-effects model we just fit, so we should be careful in interpreting those results.

Meanwhile, our robust regression results still stand, as long as we are careful about the interpretation. The correct interpretation is that, if the data collection were repeated (on women sampled the same way as in the original sample), and if we were to refit the model, 95% of the time we would expect the estimated coefficient on tenure to be in the range $[0.036, 0.042]$.

Even with robust regression, we must be careful about going beyond that statement. Here the Hausman test is probably picking up something that differs within and between person, which would cast doubt on our robust regression model in terms of interpreting $[0.036, 0.042]$ to contain the rate of return for keeping a job, economywide, for all women, without exception.

◁

Weighted regression

`regress` can perform weighted and unweighted regression. We indicate the weight by specifying the [*weight*] qualifier. By default, `regress` assumes analytic weights; see the technical note below.

▷ Example 8: Using means as regression variables

We have census data recording the death rate (`drate`) and median age (`medage`) for each state. The data also record the region of the country in which each state is located and the overall population of the state:

```
. use http://www.stata-press.com/data/r13/census9
(1980 Census data by state)

. describe

Contains data from http://www.stata-press.com/data/r13/census9.dta
  obs:            50                          1980 Census data by state
 vars:             6                          6 Apr 2013 15:43
 size:         1,450
--------------------------------------------------------------------------------
              storage   display    value
variable name   type    format     label      variable label
--------------------------------------------------------------------------------
state           str14   %-14s                 State
state2          str2    %-2s                  Two-letter state abbreviation
drate           float   %9.0g                 Death Rate
pop             long    %12.0gc               Population
medage          float   %9.2f                 Median age
region          byte    %-8.0g     cenreg     Census region
--------------------------------------------------------------------------------
Sorted by:
```

We can use factor variables to include dummy variables for region. Because the variables in the regression reflect means rather than individual observations, the appropriate method of estimation is analytically weighted least squares (Davidson and MacKinnon 2004, 261–262), where the weight is total population:

```
. regress drate medage i.region [w=pop]
(analytic weights assumed)
(sum of wgt is   2.2591e+08)

      Source |       SS           df       MS      Number of obs   =        50
-------------+----------------------------------   F(  4,     45)  =     37.21
       Model |  4096.6093         4  1024.15232   Prob > F        =    0.0000
    Residual |  1238.40987        45  27.5202192   R-squared       =    0.7679
-------------+----------------------------------   Adj R-squared   =    0.7472
       Total |  5335.01916        49  108.877942   Root MSE        =     5.246

------------------------------------------------------------------------------
       drate |      Coef.   Std. Err.      t    P>|t|     [95% Conf. Interval]
-------------+----------------------------------------------------------------
      medage |   4.283183   .5393329     7.94   0.000     3.196911    5.369455
             |
      region |
    N Cntrl  |   .3138738   2.456431     0.13   0.899    -4.633632     5.26138
      South  |  -1.438452   2.320244    -0.62   0.538    -6.111663    3.234758
       West  |  -10.90629   2.681349    -4.07   0.000    -16.30681   -5.505777
             |
       _cons |  -39.14727   17.23613    -2.27   0.028    -73.86262   -4.431915
------------------------------------------------------------------------------
```

To weight the regression by population, we added the qualifier `[w=pop]` to the end of the `regress` command. Our qualifier was vague (we did not say `[aweight=pop]`), but unless told otherwise, Stata assumes analytic weights for `regress`. Stata informed us that the sum of the weight is 2.2591×10^8; there were approximately 226 million people residing in the United States according to our 1980 data. ◁

❑ Technical note

Once we fit a weighted regression, we can obtain the appropriately weighted variance–covariance matrix of the estimators using `estat vce` and perform appropriately weighted hypothesis tests using `test`.

In the weighted regression in example 8, we see that `4.region` is statistically significant but that `2.region` and `3.region` are not. We use `test` to test the joint significance of the region variables:

```
. test 2.region 3.region 4.region
 ( 1)  2.region = 0
 ( 2)  3.region = 0
 ( 3)  4.region = 0
       F(  3,    45) =    9.84
            Prob > F =    0.0000
```

The results indicate that the region variables are jointly significant.

❑

`regress` also accepts frequency weights (`fweights`). Frequency weights are appropriate when the data do not reflect cell means, but instead represent replicated observations. Specifying `aweights` or `fweights` will not change the parameter estimates, but it will change the corresponding significance levels.

For instance, if we specified `[fweight=pop]` in the weighted regression example above—which would be statistically incorrect—Stata would treat the data as if the data represented 226 million independent observations on death rates and median age. The data most certainly do not represent that—they represent 50 observations on state averages.

With `aweights`, Stata treats the number of observations on the process as the number of observations in the data. When we specify `fweights`, Stata treats the number of observations as if it were equal to the sum of the weights; see *Methods and formulas* below.

❑ Technical note

A popular request on the help line is to describe the effect of specifying `[aweight=`*exp*`]` with `regress` in terms of transformation of the dependent and independent variables. The mechanical answer is that typing

```
. regress y x1 x2 [aweight=n]
```

is equivalent to fitting the model

$$y_j\sqrt{n_j} = \beta_0\sqrt{n_j} + \beta_1 x_{1j}\sqrt{n_j} + \beta_2 x_{2j}\sqrt{n_j} + u_j\sqrt{n_j}$$

This regression will reproduce the coefficients and covariance matrix produced by the `aweighted` regression. The mean squared errors (estimates of the variance of the residuals) will, however, be different. The transformed regression reports s_t^2, an estimate of $\mathrm{Var}(u_j\sqrt{n_j})$. The `aweighted` regression reports s_a^2, an estimate of $\mathrm{Var}(u_j\sqrt{n_j}\sqrt{N/\sum_k n_k})$, where N is the number of observations. Thus

$$s_a^2 = \frac{N}{\sum_k n_k} s_t^2 = \frac{s_t^2}{\overline{n}} \tag{1}$$

The logic for this adjustment is as follows: Consider the model

$$y = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + u$$

Assume that, were this model fit on individuals, $\mathrm{Var}(u) = \sigma_u^2$, a constant. Assume that individual data are not available; what is available are averages $(\overline{y}_j, \overline{x}_{1j}, \overline{x}_{2j})$ for $j = 1, \ldots, N$, and each average is calculated over n_j observations. Then it is still true that

$$\overline{y}_j = \beta_0 + \beta_1 \overline{x}_{1j} + \beta_2 \overline{x}_{2j} + \overline{u}_j$$

where $\overline{u}_j$ is the average of n_j mean 0, variance σ_u^2 deviates and has variance $\sigma_{\overline{u}}^2 = \sigma_u^2 / n_j$. Thus multiplying through by $\sqrt{n_j}$ produces

$$\overline{y}_j \sqrt{n_j} = \beta_0 \sqrt{n_j} + \beta_1 \overline{x}_{1j} \sqrt{n_j} + \beta_2 \overline{x}_{2j} \sqrt{n_j} + \overline{u}_j \sqrt{n_j}$$

and $\mathrm{Var}(\overline{u}_j \sqrt{n_j}) = \sigma_u^2$. The mean squared error, s_t^2, reported by fitting this transformed regression is an estimate of σ_u^2. The coefficients and covariance matrix could also be obtained by `aweighted regress`. The only difference would be in the reported mean squared error, which from (1) is $\sigma_u^2 / \overline{n}$. On average, each observation in the data reflects the averages calculated over $\overline{n} = \sum_k n_k / N$ individuals, and thus this reported mean squared error is the average variance of an observation in the dataset. We can retrieve the estimate of σ_u^2 by multiplying the reported mean squared error by $\overline{n}$.

More generally, `aweights` are used to solve general heteroskedasticity problems. In these cases, we have the model

$$y_j = \beta_0 + \beta_1 x_{1j} + \beta_2 x_{2j} + u_j$$

and the variance of u_j is thought to be proportional to a_j. If the variance is proportional to a_j, it is also proportional to αa_j, where α is any positive constant. Not quite arbitrarily, but with no loss of generality, we could choose $\alpha = \sum_k (1/a_k)/N$, the average value of the inverse of a_j. We can then write $\mathrm{Var}(u_j) = k \alpha a_j \sigma^2$, where k is the constant of proportionality that is no longer a function of the scale of the weights.

Dividing this regression through by the $\sqrt{a_j}$,

$$y_j / \sqrt{a_j} = \beta_0 / \sqrt{a_j} + \beta_1 x_{1j} / \sqrt{a_j} + \beta_2 x_{2j} / \sqrt{a_j} + u_j / \sqrt{a_j}$$

produces a model with $\mathrm{Var}(u_j / \sqrt{a_j}) = k \alpha \sigma^2$, which is the constant part of $\mathrm{Var}(u_j)$. This variance is a function of α, the average of the reciprocal weights; if the weights are scaled arbitrarily, then so is this variance.

We can also fit this model by typing

```
. regress y x1 x2 [aweight=1/a]
```

This input will produce the same estimates of the coefficients and covariance matrix; the reported mean squared error is, from (1), $\left\{N / \sum_k (1/a_k)\right\} k \alpha \sigma^2 = k \sigma^2$. This variance is independent of the scale of a_j.

❑

Instrumental variables and two-stage least-squares regression

An alternate syntax for `regress` can be used to produce instrumental-variables (two-stage least squares) estimates.

`regress` *depvar* [*varlist*$_1$ [(*varlist*$_2$)]] [*if*] [*in*] [*weight*] [, *regress_options*]

This syntax is used mainly by programmers developing estimators using the instrumental-variables estimates as intermediate results. `ivregress` is normally used to directly fit these models; see [R] **ivregress**.

With this syntax, `regress` fits a structural equation of *depvar* on *varlist*$_1$ using instrumental variables regression; (*varlist*$_2$) indicates the list of instrumental variables. With the exception of `vce(hc2)` and `vce(hc3)`, all standard `regress` options are allowed.

Video example

Simple linear regression in Stata

Stored results

`regress` 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(ll)`	log likelihood under additional assumption of i.i.d. normal errors
`e(ll_0)`	log likelihood, constant-only model
`e(N_clust)`	number of clusters
`e(rank)`	rank of `e(V)`
Macros	
`e(cmd)`	`regress`
`e(cmdline)`	command as typed
`e(depvar)`	name of dependent variable
`e(model)`	`ols` or `iv`
`e(wtype)`	weight type
`e(wexp)`	weight expression
`e(title)`	title in estimation output when `vce()` is not `ols`
`e(clustvar)`	name of cluster variable
`e(vce)`	*vcetype* specified in `vce()`
`e(vcetype)`	title used to label Std. Err.
`e(properties)`	`b V`
`e(estat_cmd)`	program used to implement `estat`
`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
`e(V_modelbased)`	model-based variance
Functions	
`e(sample)`	marks estimation sample

Methods and formulas

Methods and formulas are presented under the following headings:

Coefficient estimation and ANOVA table

Variables printed in lowercase and not boldfaced (for example, x) are scalars. Variables printed in lowercase and boldfaced (for example, $\mathbf{x}$) are column vectors. Variables printed in uppercase and boldfaced (for example, $\mathbf{X}$) are matrices.

Let $\mathbf{v}$ be a column vector of weights specified by the user. If no weights are specified, $\mathbf{v} = \mathbf{1}$. Let $\mathbf{w}$ be a column vector of normalized weights. If no weights are specified or if the user specified `fweights` or `iweights`, $\mathbf{w} = \mathbf{v}$. Otherwise, $\mathbf{w} = \big\{\mathbf{v}/(\mathbf{1}'\mathbf{v})\big\}(\mathbf{1}'\mathbf{1})$.

The number of observations, n, is defined as $\mathbf{1}'\mathbf{w}$. For `iweights`, this is truncated to an integer. The sum of the weights is $\mathbf{1}'\mathbf{v}$. Define $c = 1$ if there is a constant in the regression and zero otherwise. Define k as the number of right-hand-side variables (including the constant).

Let $\mathbf{X}$ denote the matrix of observations on the right-hand-side variables, $\mathbf{y}$ the vector of observations on the left-hand-side variable, and $\mathbf{Z}$ the matrix of observations on the instruments. If the user specifies no instruments, then $\mathbf{Z} = \mathbf{X}$. In the following formulas, if the user specifies weights, then $\mathbf{X}'\mathbf{X}$, $\mathbf{X}'\mathbf{y}$, $\mathbf{y}'\mathbf{y}$, $\mathbf{Z}'\mathbf{Z}$, $\mathbf{Z}'\mathbf{X}$, and $\mathbf{Z}'\mathbf{y}$ are replaced by $\mathbf{X}'\mathbf{D}\mathbf{X}$, $\mathbf{X}'\mathbf{D}\mathbf{y}$, $\mathbf{y}'\mathbf{D}\mathbf{y}$, $\mathbf{Z}'\mathbf{D}\mathbf{Z}$, $\mathbf{Z}'\mathbf{D}\mathbf{X}$, and $\mathbf{Z}'\mathbf{D}\mathbf{y}$, respectively, where $\mathbf{D}$ is a diagonal matrix whose diagonal elements are the elements of $\mathbf{w}$. We suppress the $\mathbf{D}$ below to simplify the notation.

If no instruments are specified, define $\mathbf{A}$ as $\mathbf{X}'\mathbf{X}$ and $\mathbf{a}$ as $\mathbf{X}'\mathbf{y}$. Otherwise, define $\mathbf{A}$ as $\mathbf{X}'\mathbf{Z}(\mathbf{Z}'\mathbf{Z})^{-1}(\mathbf{X}'\mathbf{Z})'$ and $\mathbf{a}$ as $\mathbf{X}'\mathbf{Z}(\mathbf{Z}'\mathbf{Z})^{-1}\mathbf{Z}'\mathbf{y}$.

The coefficient vector $\mathbf{b}$ is defined as $\mathbf{A}^{-1}\mathbf{a}$. Although not shown in the notation, unless `hascons` is specified, $\mathbf{A}$ and $\mathbf{a}$ are accumulated in deviation form and the constant is calculated separately. This comment applies to all statistics listed below.

The total sum of squares, TSS, equals $\mathbf{y}'\mathbf{y}$ if there is no intercept and $\mathbf{y}'\mathbf{y} - \big\{(\mathbf{1}'\mathbf{y})^2/n\big\}$ otherwise. The degrees of freedom is $n - c$.

The error sum of squares, ESS, is defined as $\mathbf{y}'\mathbf{y} - 2\mathbf{b}\mathbf{X}'\mathbf{y} + \mathbf{b}'\mathbf{X}'\mathbf{X}\mathbf{b}$ if there are instruments and as $\mathbf{y}'\mathbf{y} - \mathbf{b}'\mathbf{X}'\mathbf{y}$ otherwise. The degrees of freedom is $n - k$.

The model sum of squares, MSS, equals TSS − ESS. The degrees of freedom is $k - c$.

The mean squared error, s^2, is defined as ESS$/(n - k)$. The root mean squared error is s, its square root.

The F statistic with $k - c$ and $n - k$ degrees of freedom is defined as

$$F = \frac{\text{MSS}}{(k-c)s^2}$$

if no instruments are specified. If instruments are specified and $c = 1$, then F is defined as

$$F = \frac{(\mathbf{b}-\mathbf{c})'\mathbf{A}(\mathbf{b}-\mathbf{c})}{(k-1)s^2}$$

where $\mathbf{c}$ is a vector of $k - 1$ zeros and kth element $\mathbf{1}'\mathbf{y}/n$. Otherwise, F is defined as "missing". (Here you may use the `test` command to construct any F test that you wish.)

The R-squared, R^2, is defined as $R^2 = 1 - \text{ESS}/\text{TSS}$.

The adjusted R-squared, R_a^2, is $1 - (1 - R^2)(n - c)/(n - k)$.

If `vce(robust)` is not specified, the conventional estimate of variance is $s^2\mathbf{A}^{-1}$. The handling of `vce(robust)` is described below.

A general notation for the robust variance calculation

Put aside all context of linear regression and the notation that goes with it—we will return to it. First, we are going to establish a notation for describing robust variance calculations.

The calculation formula for the robust variance calculation is

$$\widehat{\mathcal{V}} = q_c \widehat{\mathbf{V}} \Bigg(\sum_{k=1}^{M} \mathbf{u}_k^{(G)\prime} \mathbf{u}_k^{(G)} \Bigg) \widehat{\mathbf{V}}$$

where

$$\mathbf{u}_k^{(G)} = \sum_{j \in G_k} w_j \mathbf{u}_j$$

$G_1, G_2, \dots, G_M$ are the clusters specified by `vce(cluster` *clustvar*`)`, and w_j are the user-specified weights, normalized if `aweights` or `pweights` are specified and equal to 1 if no weights are specified.

For `fweights` without clusters, the variance formula is

$$\widehat{\mathcal{V}} = q_c \widehat{\mathbf{V}} \Bigg(\sum_{j=1}^{N} w_j \mathbf{u}_j' \mathbf{u}_j \Bigg) \widehat{\mathbf{V}}$$

which is the same as expanding the dataset and making the calculation on the unweighted data.

If `vce(cluster` *clustvar*`)` is not specified, $M = N$, and each cluster contains 1 observation. The inputs into this calculation are

- $\widehat{\mathbf{V}}$, which is typically a conventionally calculated variance matrix;
- $\mathbf{u}_j$, $j = 1, \dots, N$, a row vector of scores; and
- q_c, a constant finite-sample adjustment.

Thus we can now describe how estimators apply the robust calculation formula by defining $\widehat{\mathbf{V}}$, $\mathbf{u}_j$, and q_c.

Two definitions are popular enough for q_c to deserve a name. The regression-like formula for q_c (Fuller et al. 1986) is

$$q_c = \frac{N-1}{N-k} \, \frac{M}{M-1}$$

where M is the number of clusters and N is the number of observations. For weights, N refers to the sum of the weights if weights are frequency weights and the number of observations in the dataset (ignoring weights) in all other cases. Also note that, weighted or not, $M = N$ when `vce(cluster` *clustvar*`)` is not specified, and then $q_c = N/(N - k)$.

The asymptotic-like formula for q_c is

$$q_c = \frac{M}{M-1}$$

where $M = N$ if `vce(cluster` *clustvar*`)` is not specified.

See [U] 20.21 Obtaining robust variance estimates and [P] _robust for a discussion of the robust variance estimator and a development of these formulas.

Robust calculation for regress

For regress, $\widehat{\mathbf{V}} = \mathbf{A}^{-1}$. The other terms are

No instruments, vce(robust), but not vce(hc2) or vce(hc3),

$$\mathbf{u}_j = (y_j - \mathbf{x}_j\mathbf{b})\mathbf{x}_j$$

and $q_{\rm c}$ is given by its regression-like definition.

No instruments, vce(hc2),

$$\mathbf{u}_j = \frac{1}{\sqrt{1-h_{jj}}}(y_j - \mathbf{x}_j\mathbf{b})\mathbf{x}_j$$

where $q_{\rm c} = 1$ and $h_{jj} = \mathbf{x}_j(\mathbf{X}'\mathbf{X})^{-1}\mathbf{x}_j{}'$.

No instruments, vce(hc3),

$$\mathbf{u}_j = \frac{1}{1-h_{jj}}(y_j - \mathbf{x}_j\mathbf{b})\mathbf{x}_j$$

where $q_{\rm c} = 1$ and $h_{jj} = \mathbf{x}_j(\mathbf{X}'\mathbf{X})^{-1}\mathbf{x}_j{}'$.

Instrumental variables,

$$\mathbf{u}_j = (y_j - \mathbf{x}_j\mathbf{b})\widehat{\mathbf{x}}_j$$

where $q_{\rm c}$ is given by its regression-like definition, and

$$\widehat{\mathbf{x}}_j' = \mathbf{P}\mathbf{z}_j{}'$$

where $\mathbf{P} = (\mathbf{X}'\mathbf{Z})(\mathbf{Z}'\mathbf{Z})^{-1}$.

Acknowledgments

The robust estimate of variance was first implemented in Stata by Mead Over of the Center for Global Development, Dean Jolliffe of the World Bank, and Andrew Foster of the Department of Economics at Brown University (Over, Jolliffe, and Foster 1996).

The history of regression is long and complicated: the books by Stigler (1986) and Hald (1998) are devoted largely to the story. Legendre published first on least squares in 1805. Gauss published later in 1809, but he had the idea earlier. Gauss, and especially Laplace, tied least squares to a normal errors assumption. The idea of the normal distribution can itself be traced back to De Moivre in 1733. Laplace discussed a variety of other estimation methods and error assumptions over his long career, while linear models long predate either innovation. Most of this work was linked to problems in astronomy and geodesy.

A second wave of ideas started when Galton used graphical and descriptive methods on data bearing on heredity to develop what he called regression. His term reflects the common phenomenon that characteristics of offspring are positively correlated with those of parents but with regression slope such that offspring "regress toward the mean". Galton's work was rather intuitive: contributions from Pearson, Edgeworth, Yule, and others introduced more formal machinery, developed related ideas on correlation, and extended application into the biological and social sciences. So most of the elements of regression as we know it were in place by 1900.

Pierre-Simon Laplace (1749–1827) was born in Normandy and was early recognized as a remarkable mathematician. He weathered a changing political climate well enough to rise to Minister of the Interior under Napoleon in 1799 (although only for 6 weeks) and to be made a Marquis by Louis XVIII in 1817. He made many contributions to mathematics and physics, his two main interests being theoretical astronomy and probability theory (including statistics). Laplace transforms are named for him.

Adrien-Marie Legendre (1752–1833) was born in Paris (or possibly in Toulouse) and educated in mathematics and physics. He worked in number theory, geometry, differential equations, calculus, function theory, applied mathematics, and geodesy. The Legendre polynomials are named for him. His main contribution to statistics is as one of the discoverers of least squares. He died in poverty, having refused to bow to political pressures.

Johann Carl Friedrich Gauss (1777–1855) was born in Braunschweig (Brunswick), now in Germany. He studied there and at Göttingen. His doctoral dissertation at the University of Helmstedt was a discussion of the fundamental theorem of algebra. He made many fundamental contributions to geometry, number theory, algebra, real analysis, differential equations, numerical analysis, statistics, astronomy, optics, geodesy, mechanics, and magnetism. An outstanding genius, Gauss worked mostly in isolation in Göttingen.

Francis Galton (1822–1911) was born in Birmingham, England, into a well-to-do family with many connections: he and Charles Darwin were first cousins. After an unsuccessful foray into medicine, he became independently wealthy at the death of his father. Galton traveled widely in Europe, the Middle East, and Africa, and became celebrated as an explorer and geographer. His pioneering work on weather maps helped in the identification of anticyclones, which he named. From about 1865, most of his work was centered on quantitative problems in biology, anthropology, and psychology. In a sense, Galton (re)invented regression, and he certainly named it. Galton also promoted the normal distribution, correlation approaches, and the use of median and selected quantiles as descriptive statistics. He was knighted in 1909.

References

Adkins, L. C., and R. C. Hill. 2011. *Using Stata for Principles of Econometrics*. 4th ed. Hoboken, NJ: Wiley.

Alexandersson, A. 1998. gr32: Confidence ellipses. *Stata Technical Bulletin* 46: 10–13. Reprinted in *Stata Technical Bulletin Reprints*, vol. 8, pp. 54–57. College Station, TX: Stata Press.

Angrist, J. D., and J.-S. Pischke. 2009. *Mostly Harmless Econometrics: An Empiricist's Companion*. Princeton, NJ: Princeton University Press.

Becketti, S. 2013. *Introduction to Time Series Using Stata*. College Station, TX: Stata Press.

Cameron, A. C., and P. K. Trivedi. 2010. *Microeconometrics Using Stata*. Rev. ed. College Station, TX: Stata Press.

Chatterjee, S., and A. S. Hadi. 2012. *Regression Analysis by Example*. 5th ed. New York: Hoboken, NJ.

Davidson, R., and J. G. MacKinnon. 1993. *Estimation and Inference in Econometrics*. New York: Oxford University Press.

——. 2004. *Econometric Theory and Methods*. New York: Oxford University Press.

Dohoo, I., W. Martin, and H. Stryhn. 2010. *Veterinary Epidemiologic Research*. 2nd ed. Charlottetown, Prince Edward Island: VER Inc.

——. 2012. *Methods in Epidemiologic Research*. Charlottetown, Prince Edward Island: VER Inc.

Draper, N., and H. Smith. 1998. *Applied Regression Analysis*. 3rd ed. New York: Wiley.

Dunnington, G. W. 1955. *Gauss: Titan of Science*. New York: Hafner Publishing.

Duren, P. 2009. Changing faces: The mistaken portrait of Legendre. *Notices of the American Mathematical Society* 56: 1440–1443.

Filoso, V. 2013. Regression anatomy, revealed. *Stata Journal* 13: 92–106.

Fuller, W. A., W. J. Kennedy, Jr., D. Schnell, G. Sullivan, and H. J. Park. 1986. *PC CARP*. Software package. Ames, IA: Statistical Laboratory, Iowa State University.

Gillham, N. W. 2001. *A Life of Sir Francis Galton: From African Exploration to the Birth of Eugenics*. New York: Oxford University Press.

Gillispie, C. C. 1997. *Pierre-Simon Laplace, 1749–1827: A Life in Exact Science*. Princeton: Princeton University Press.

Gould, W. W. 2011a. Understanding matrices intuitively, part 1. The Stata Blog: Not Elsewhere Classified. http://blog.stata.com/2011/03/03/understanding-matrices-intuitively-part-1/.

——. 2011b. Use poisson rather than regress; tell a friend. The Stata Blog: Not Elsewhere Classified. http://blog.stata.com/2011/08/22/use-poisson-rather-than-regress-tell-a-friend/.

Greene, W. H. 2012. *Econometric Analysis*. 7th ed. Upper Saddle River, NJ: Prentice Hall.

Hald, A. 1998. *A History of Mathematical Statistics from 1750 to 1930*. New York: Wiley.

Hamilton, L. C. 2013. *Statistics with Stata: Updated for Version 12*. 8th ed. Boston: Brooks/Cole.

Hill, R. C., W. E. Griffiths, and G. C. Lim. 2011. *Principles of Econometrics*. 4th ed. Hoboken, NJ: Wiley.

Kmenta, J. 1997. *Elements of Econometrics*. 2nd ed. Ann Arbor: University of Michigan Press.

Kohler, U., and F. Kreuter. 2012. *Data Analysis Using Stata*. 3rd ed. College Station, TX: Stata Press.

Long, J. S., and J. Freese. 2000. sg152: Listing and interpreting transformed coefficients from certain regression models. *Stata Technical Bulletin* 57: 27–34. Reprinted in *Stata Technical Bulletin Reprints*, vol. 10, pp. 231–240. College Station, TX: Stata Press.

MacKinnon, J. G., and H. L. White, Jr. 1985. Some heteroskedasticity-consistent covariance matrix estimators with improved finite sample properties. *Journal of Econometrics* 29: 305–325.

Mitchell, M. N. 2012. *Interpreting and Visualizing Regression Models Using Stata*. College Station, TX: Stata Press.

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

Over, M., D. Jolliffe, and A. Foster. 1996. sg46: Huber correction for two-stage least squares estimates. *Stata Technical Bulletin* 29: 24–25. Reprinted in *Stata Technical Bulletin Reprints*, vol. 5, pp. 140–142. College Station, TX: Stata Press.

Peracchi, F. 2001. *Econometrics*. Chichester, UK: Wiley.

Plackett, R. L. 1972. Studies in the history of probability and statistics: XXIX. The discovery of the method of least squares. *Biometrika* 59: 239–251.

Rogers, W. H. 1991. smv2: Analyzing repeated measurements—some practical alternatives. *Stata Technical Bulletin* 4: 10–16. Reprinted in *Stata Technical Bulletin Reprints*, vol. 1, pp. 123–131. College Station, TX: Stata Press.

Royston, P., and G. Ambler. 1998. sg79: Generalized additive models. *Stata Technical Bulletin* 42: 38–43. Reprinted in *Stata Technical Bulletin Reprints*, vol. 7, pp. 217–224. College Station, TX: Stata Press.

Schonlau, M. 2005. Boosted regression (boosting): An introductory tutorial and a Stata plugin. *Stata Journal* 5: 330–354.

Stigler, S. M. 1986. *The History of Statistics: The Measurement of Uncertainty before 1900.* Cambridge, MA: Belknap Press.

Tyler, J. H. 1997. sg73: Table making programs. *Stata Technical Bulletin* 40: 18–23. Reprinted in *Stata Technical Bulletin Reprints*, vol. 7, pp. 186–192. College Station, TX: Stata Press.

Weesie, J. 1998. sg77: Regression analysis with multiplicative heteroscedasticity. *Stata Technical Bulletin* 42: 28–32. Reprinted in *Stata Technical Bulletin Reprints*, vol. 7, pp. 204–210. College Station, TX: Stata Press.

Weisberg, S. 2005. *Applied Linear Regression*. 3rd ed. New York: Wiley.

Wooldridge, J. M. 2010. *Econometric Analysis of Cross Section and Panel Data*. 2nd ed. Cambridge, MA: MIT Press.

——. 2013. *Introductory Econometrics: A Modern Approach*. 5th ed. Mason, OH: South-Western.

Zimmerman, F. 1998. sg93: Switching regressions. *Stata Technical Bulletin* 45: 30–33. Reprinted in *Stata Technical Bulletin Reprints*, vol. 8, pp. 183–186. College Station, TX: Stata Press.

Also see

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

[R] **regress postestimation diagnostic plots** — Postestimation plots for regress

[R] **regress postestimation time series** — Postestimation tools for regress with time series

[R] **anova** — Analysis of variance and covariance

[R] **contrast** — Contrasts and linear hypothesis tests after estimation

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

[SEM] **example 6** — Linear regression

[SEM] **intro 5** — Tour of models

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

[TS] **forecast** — Econometric model forecasting

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